VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF TELECOMMUNICATIONS

WEBOVÁ VIZUALIZACE KRYPTOGRAFICKÝCH SYSTÉMŮ

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

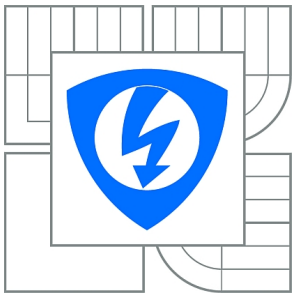
AUTOR PRÁCE
AUTHOR

MAREK SIKORA

BRNO 2015

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF TELECOMMUNICATIONS

WEBOVÁ VIZUALIZACE KRYPTOGRAFICKÝCH SYSTÉMŮ

WEB VISUALISATION OF CRYPTOGRAPHIC SYSTEMS

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

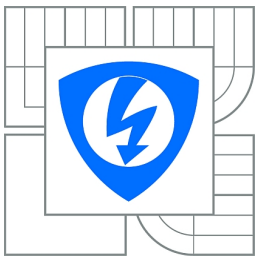
AUTOR PRÁCE — MAREK SIKORA

AUTHOR

VEDOUCÍ PRÁCE — Ing. JAN HAJNÝ, Ph.D.

SUPERVISOR

BRNO 2015



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Bakalářská práce

bakalářský studijní obor
Teleinformatika

Student:	Marek Sikora	***ID:***	155135
Ročník:	3	***Akademický rok:***	2014/2015

NÁZEV TÉMATU:

Webová vizualizace kryptografických systémů

POKYNY PRO VYPRACOVÁNÍ:

Cílem projektu je vytvořit webovou vizualizaci a demonstrátor kryptografického systému. Použité technologie (HTML5, Flash, Java, apod.) si může student zvolit, výsledkem je webová stránka demonstrující zadaný systém založený na protokolu HM12 či HM14. Důležitým aspektem je interaktivita vizualizace a jasná demonstrace základních kryptografických principů. Výsledkem práce je implementace všech protokolů atributového autentizačního systému pomocí webových technologií.

DOPORUČENÁ LITERATURA:

[1] STALLINGS, William. Cryptography and network security: principles and practice. Seventh edition. xix, 731 pages. ISBN 01-333-5469-5.
[2] HAJNÝ, J.; MALINA, L. Unlinkable Attribute-Based Credentials with Practical Revocation on Smart-Cards. In Smart Card Research and Advanced Applications. Lecture Notes in Computer Science. LNCS. Berlin: Springer- Verlag, 2013. s. 62-76. ISBN: 978-3-642-37287- 2. ISSN: 0302- 9743.
[3] CASTRO, Elizabeth a Bruce HYSLOP. HTML5 a CSS3: názorný průvodce tvorbou WWW stránek. 1. vyd. Brno: Computer Press, 2012, 439 s. ISBN 978-80-251-3733-8.

Termín zadání:	9.2.2015	***Termín odevzdání:***	2.6.2015

Vedoucí práce: Ing. Jan Hajný, Ph.D.
Konzultanti bakalářské práce:

doc. Ing. Jiří Mišurec, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

ABSTRAKT

Tato práce se zabývá webovými technologiemi pro interaktivní vizualizaci obsahu. Rozebírá především novou specifikaci jazyka HTML5, jeho možnosti animací pomocí jazyka Javascript a jejich praktické využití při tvorbě webových stránek. Dále se práce zaměřuje na vývojový nástroj Adobe Edge Animate, pomocí kterého je návrh stránky realizován. Práce se dále zabývá kryptografickými systémy a protokolem HM12. Zobrazuje základní přehled a funkčnost tohoto protokolu. V poslední části práce je pak popsána vlastní implementace těchto dvou témat, a to vytvoření webové vizualizace – demonstrátoru daného kryptografického systému včetně vytvoření plně funkčního modelu.

KLÍČOVÁ SLOVA

webové aplikace, webové animace, HTML5, Javascript, BigInteger, bezpečnost, kryptografie, HM12, ověřování atributů, anonymní ověření, vizualizace

ABSTRACT

This thesis deals with web technologies for interactive visualization of content. It discusses the new HTML5 specification, its animation possibilities using Javascript and their practical application for creating web pages. Furthermore, the work focuses on the development tool Adobe Edge Animate, through which we realize page design. The study also discusses cryptographic systems and protocol HM12. It displays a basic overview of the functionality of this Protocol. In the last part of the thesis describes the actual implementation of these two topics, namely the creation of web visualization – demonstrator of the cryptographic system including a fully operational model.

KEYWORDS

web application, web animation, HTML5, Javascript, BigInteger, security, cryptography, HM12, verification of attributes, anonymous authentication

SIKORA, Marek *Webová vizualizace kryptografických systémů*: bakalářská práce. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2014. 38 s. Vedoucí práce byl Ing. Jan Hajný, Ph.D.

PROHLÁŠENÍ

Prohlašuji, že svou bakalářskou práci na téma „Webová vizualizace kryptografických systémů“ jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

(podpis autora)

SIX research centre
sensor, information and communication systems

Faculty of Electrical Engineering
and Communication

Brno University of Technology
Technicka 12, CZ-61600 Brno, Czechia

http://www.six.feec.vutbr.cz

Výzkum popsaný v této bakalářské práci byl realizovaný v laboratořích podpořených projektem Centrum senzorických, informačních a komunikačních systémů (SIX); registrační číslo CZ.1.05/2.1.00/03.0072, operačního programu Výzkum a vývoj pro inovace.

MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

EVROPSKÁ UNIE
EVROPSKÝ FOND PRO REGIONÁLNÍ ROZVOJ
INVESTICE DO VAŠÍ BUDOUCNOSTI

2007–13
OP Výzkum a vývoj pro inovace

OBSAH

SEZNAM OBRÁZKŮ

ÚVOD

Tato práce se zabývá tvorbou interaktivní webové vizualizace pomocí moderních webových technologií. Je zaměřena především na nejnovější specifikaci značkovacího jazyka HTML5 a jejího praktického využití při tvorbě interaktivního animovaného webu. Cílem této práce je využití vybrané technologie k vytvoření webové stránky – demonstrátoru kryptografického protokolu HM12.

V první části se pojednává obecně o webových technologiích a poté je zde představen preferovaný jazyk HTML5 společně s Javascriptem, který je nedílnou součástí webových animací.

Dále je zde podrobně představen vývojový nástroj Adobe Edge Animate, který je využit k vytvoření webové stránky. Jsou zde rozepsány základní informace, přehled uživatelského prostředí a podrobnější popis funkce.

Druhá část pojednává o kryptografických systémech obecně a o kryptografickém protokolu HM12 určeného k demonstrování ve vytvořeném interaktivním prostředí. Jsou zde popsány základní informace, a poté podrobnější schéma fungování tohoto protokolu.

Ve třetí části práce rozebírá především implementaci funkčního modelu kryptografického protokolu HM12 s technologii HTML5 a Javascript. Jsou zde řešeny výpočty s velkými čísly o velikosti až 1024 bitů, výpočty kryptografických klíčů na stranách jednotlivých entit a v poslední řadě prezentace dat, vytváření webového prostředí a animací bez použití dalších technologií (Java nebo Flash).

Poslední čtvrtá část popisuje spuštění vytvořených webových stránek na webovém serveru. Je zde krok za krokem rozebrán postup instalace webového serveru Apache na serverovém počítači s operačním systémem Debian.

1 WEBOVÉ TECHNOLOGIE

Základem každé webové aplikace nebo webových stránek je samostatný HTML dokument. Tento dokument je možné pomocí webového prohlížeče zobrazit na monitoru počítače či displeji mobilního přístroje. Informace jsou prezentovány ve formě hypertextu, který je vytvořen značkovacím jazykem HTML nebo XHTML.

Webové stránky mohou být uloženy v podobě souborů na pevném disku nebo je poskytují webové servery prostřednictvím počítačové sítě nebo Internetu, kde jsou přenášeny pomocí protokolu HTTP [14].

Zpočátku byly webové stránky pouze v podobě statického textu. Nejčastěji to byly odborné články a firemní weby se základními informacemi o firmě. Obrázků bylo poskrovnu. Postupem času a vývoje začali tvůrci webových stránek používat své stránky i pro komunikaci s návštěvníky, např. pomocí jednoduchých formulářů. Ke komunikaci se začaly používat stále složitější a rozsáhlejší skripty. Internet se stával dynamičtější.

Pro přidání dynamických prvků do uživatelského rozhraní se používá skriptovací jazyk, nejčastěji JavaScript. Dalším nástrojem, kterým je možné vytvářet dynamický obsah webových stránek jsou aplety. Jedná se o malé aplikace psané v populárním jazyce Java, které jsou vkládány do těla HTML stránky a pomocí virtuálního stroje JVM (Java Virtual Machine) pouštěny v prohlížeči, který podporuje Javu. Další technologie, sloužící převážně grafikům, kteří se soustředí na vizuální stránku webu, nese označení Flash. Slouží nejen ke tvorbě animací, ale pomocí skriptovacího jazyka Action Script je možné vytvářet grafické aplikace, například hry [3].

Webový prohlížeč v průběhu běhu aplikace interpretuje a zobrazuje stránky a funguje jako univerzální klient pro libovolnou webovou aplikaci. Specifikace nové verze jazyka HTML - HTML5 obsahuje řadu rozšíření zaměřených právě na tvorbu webových aplikací.

V této práci se pro tvorbu kompletního uživatelského rozhraní využívá právě jazyka HMTL5 kvůli jeho aktuálnosti a jednoduché implementaci.

1.1 HMTL 5

HTML5 je verze značkovacího jazyka HTML sloužícího pro tvorbu webových stránek. Finální specifikace byla vydána 28. října 2014. Proti předchozí verzi HTML4 z roku 1997 přináší podstatné změny, přičemž mezi nejdůležitější patří [11]:

- nová sémantika a zvýšení přehlednosti zdrojového kódu,
- přímá podpora přehrávání a vkládání multimédií,
- podpora pro aplikace, které fungují i offline,
- interpretace vektorové grafiky a kreslení prostřednictvím skriptovacích jazyků,
- spousta dalších nových funkcí, značek a atributů.

HTML 5 ve spojení s JavaScriptem také nabízí tvorbu pokročilých animací u animovaných banerů, uživatelských prostředí či animovaných videoklipů. Díky tomu dokáže HTML5 zcela nahradit používání zastaralé technologie Flash i vkládání apletů jazyku Java.

1.2 JavaScript

1.2.1 O JavaScriptu

JavaScript je multiplatformní, objektově orientovaný skriptovací jazyk. Zpravidla se používá jako interpretovaný programovací jazyk pro WWW stránky, často vkládaný přímo do HTML kódu stránky. Jsou jím obvykle ovládány různé interaktivní prvky GUI (tlačítka, textová políčka) nebo tvořeny animace a efekty obrázků.

Jeho syntaxe patří do rodiny jazyků C/C++/Java. Slovo Java je však součástí jeho názvu pouze z marketingových důvodů a s programovacím jazykem Java jej vedle názvu spojuje jen podobná syntaxe [12].

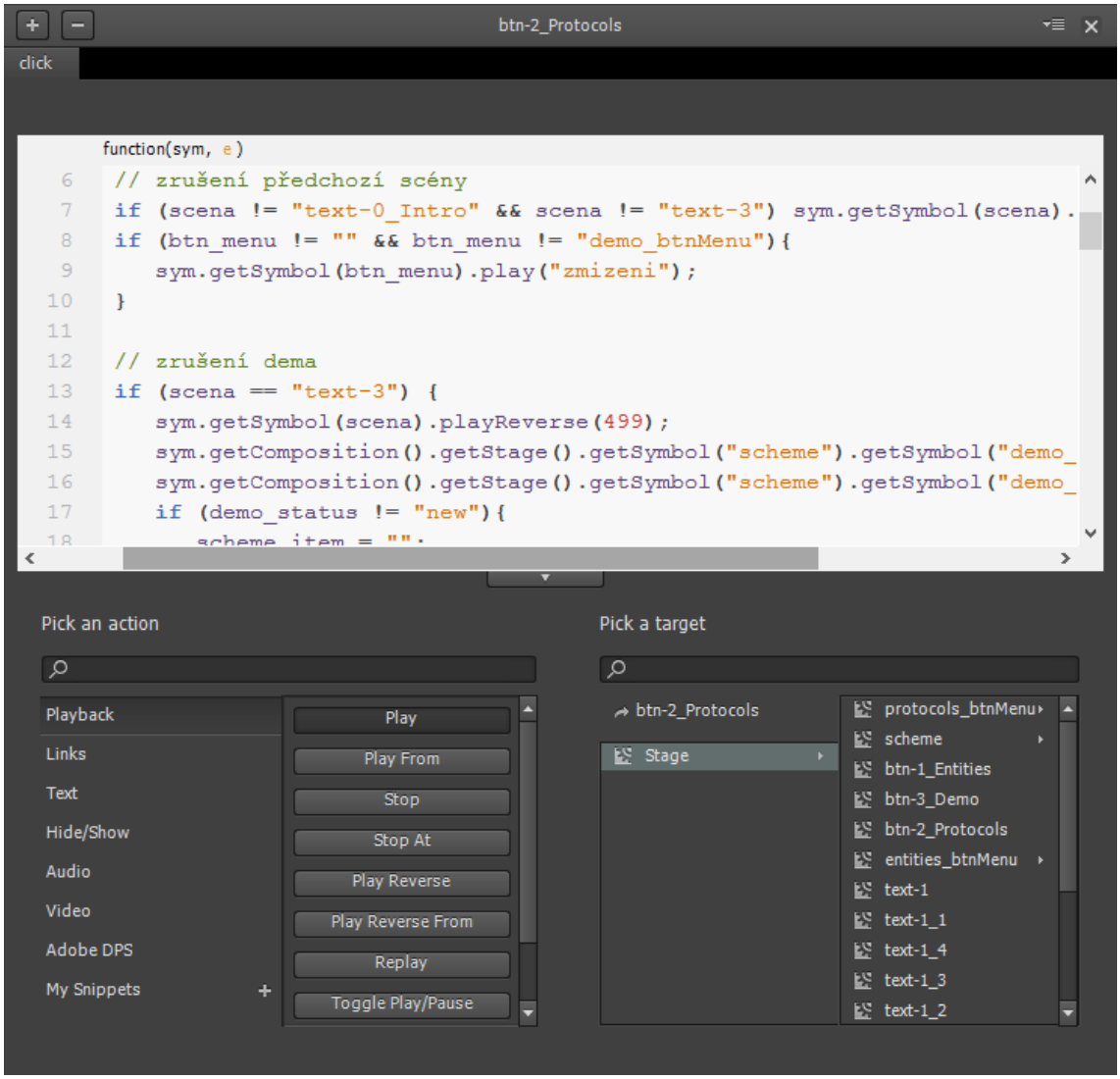
Program v JavaScriptu se obvykle spouští až po stažení WWW stránky z Internetu (tzv. na straně klienta), na rozdíl od ostatních jiných programovacích jazyků (např. PHP a ASP), které se spouštějí na straně serveru ještě před stažením z Internetu. Z toho plynou jistá bezpečností omezení, JavaScript např. nemůže pracovat se soubory, aby tím neohrozil soukromí uživatele.

1.2.2 Použití Javascriptu

Praktický obsah této práce je zpracován skripty Javascriptu. Pomocí něj jsou vytvořeny knihovny, které jsou vloženy do `html` souboru webové stránky. Tyto knihovny definují veškeré použité animace, jejich načasování, jednotlivé objekty a akce, které se mají vykonávat při akcích návštěvníka, viz obr. 1.1. Další JS knihovny obsahují

výpočty požadovaných protokolů, definice datových typů a také hashovací funkci SHA1.

Těmto knihovnám se podrobněji věnuje část 3.2.1.



Obr. 1.1: Příklad skriptu s událostmi

1.3 Adobe Edge Animate

Nástroj Edge Animate od firmy Adobe Systems byl na trh uveden v roce 2011. Jedná se o nástroj pro webové vývojáře, pomocí kterého lze vytvářet interaktivní animace HTML5, publikovat digitální materiál, tvořit multimediální reklamy a vytvářet spoustu dalších prvků kompatibilních s prohlížeči pro stolní počítače i mobilní zařízení. Adobe prohlašuje, že Edge byl vyvinut jako nový multimediální nástroj pro

zobrazování webového obsahu v internetovém prohlížeči na základě úspěchu populární platformy Flash [9]. Tuto starší technologii postupně nahrazuje nová specifikace jazyka HTML – HTML5.

Nástroj Edge Animate využívá ke tvorbě webových stránek a animací nejnovější webové technologie, tj. HMTL5, CSS3, JavaScript a jQuery [9]. Výsledkem je účinný, ale hlavně velmi jednoduchý kód, než jak tomu bylo dříve u kombinování dalších platforem pro vytváření webových aplikací nebo animací, např. Javy nebo Flashe.

Edge Animate je editor typu WYSIWYG, což je akronym anglické věty „What you see is what you get“, česky „co vidíš, to dostaneš“. Pro práci tedy není potřeba znalosti jazyka HMTL ani Javascriptu, protože program kód vygeneruje sám. Nicméně bez těchto znalostí je vývojář poněkud omezen. Výsledkem by v takovém případě byla vcelku primitivní animovaná stránka. Proto Edge dává vývojáři možnost zasahovat přímo do kódu skriptů a ručně programovat složitější funkce.

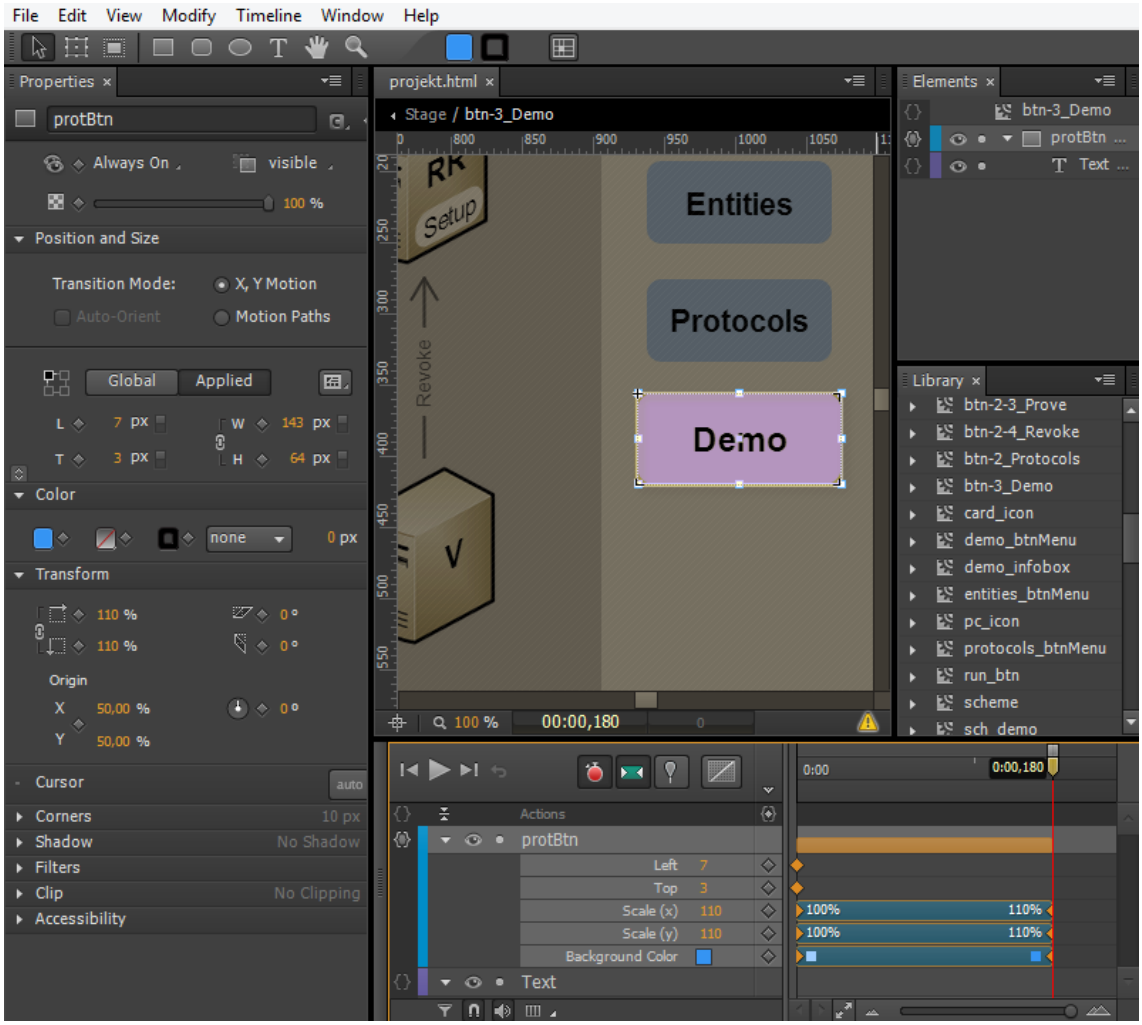
1.3.1 Práce s Edge Animate

Uživatelské prostředí Edge Animate obsahuje základní animační nástroje, které mohou být na první pohled povědomé například z Flash animátorů. Nachází se zde přehled objektů vložených na hlavním plátně, seznam symbolů, panel vlastností vybraného objektu a taky časovou osu, díky které je možné měnit v čase jednotlivé vlastnosti vybraných objektů.

Tzv. symboly tady představují seskupení více objektů se svými dílčími animacemi a vlastnostmi. Například symbol vrtulník by obsahoval nákres karoserie a nákres vrtule. Vrtule by měla definovanou animaci opakující se rotace. Při vložení symbolu vrtulník na hlavní plátno by se zobrazil rovnou celý vrtulník s otáčející se vrtulí. Celý vrtulník je pak možné na plátně rozpohybovat a udělat tak animaci jak letí.

Prvek symbol je proto důležitý stavební kámen všech animací. Kromě animací lze taky u symbolů definovat funkce, které se vykonají při nějaké akci, např. najetí kurzoru na symbol, kliknutí, atd. Toto je ideální třeba pro tvorbu animovaných tlačítek, viz obr. 1.2

Edge Animate vygeneruje základní soubor s příponou `.html`, který spouštíme v internetovém prohlížeči. Formou připojených skriptů připojí další javascriptové soubory s příponou `.js`, které obsahují definici prostředí stránky, oken, objektů a také akcí a animací. Tyto skripty generují kód do hlavní `html` stránky, čímž vytváří dynamický obsah.



Obr. 1.2: Prostředí Adobe Edge Animate

2 KRYPTOGRAFICKÉ SYSTÉMY

Bezpečnost uživatelů elektronických systémů je dnes stále důležitější hlavně díky přibývajícím službám a možnostem Internetu, nebo různých elektronických komunikačních systémů. Jedno z aktuálních témat, které nás především zajímá, je ochrana osobních dat a zamezení jejich zneužití. Dobrým příkladem jsou stále častější spekulace o zneužívání osobních dat poskytovateli služeb. Příkladem může být sledování chování uživatelů na Internetu a nalezení zvyků uživatelů pro užší cílení reklamy, popř. předávání citlivých údajů dalším subjektům za účelem cílených reklam.

Existují však mnohem závažnější příklady zneužívání citlivých dat. Poskytovatelé služeb mohou sledovat uživatelův pohyb v systému, jaké konkrétní služby uživatel používá, v jakém čase a s jakou pravidelností. Tyto údaje pak mohou o uživateli prozradit další informace, které by měly zůstat skryty.

Nejhorší případ zneužití v oblasti ochrany dat je tzv. krádež elektronické identity, kdy útočník může v elektronickém systému zcela zastoupit původního uživatele.

Problém nastává, když uživatel musí prokázat některý ze svých atributů, aby mohl pokračovat požadované činnosti. Příkladem může být člověk, který chce jít do kina na film pro dospělé. Prodavači u pokladny musí prokázat, že je starší 18 let. Prodavač se podívá na občanský průkaz a kromě věků může zjistit i přesný datum narození, jméno, přímení a bydliště. Aby čtenář ochránil své citlivé údaje, musí použít nějaký kryptografický systém, který nabízí anonymitu uživatele beze ztráty možnosti ověření, které vyžaduje prodavač.

Jedním z řešení je použití kryptografického systému prověření atributů. Soukromé údaje jsou chráněny anonymitou, odkryty jsou pouze atributy potřebné k ověření určité vlastnosti [2].

Kryptografický systém, demonstrovaný v této práci, představuje protokol HM12, který má výše uvedené vlastnosti.

2.1 Protokol HM12

Protokol HM12 patři do skupiny protokolů, které ověřují požadovaný atribut (věk, národnost, atd.) bez odhalení identity ověřovaného. K tomu se používají elektronické identifikační karty, tzv. smart–cards. Aby byla zajištěna tato anonymita, musí protokol splňovat následující vlastnosti [5]:

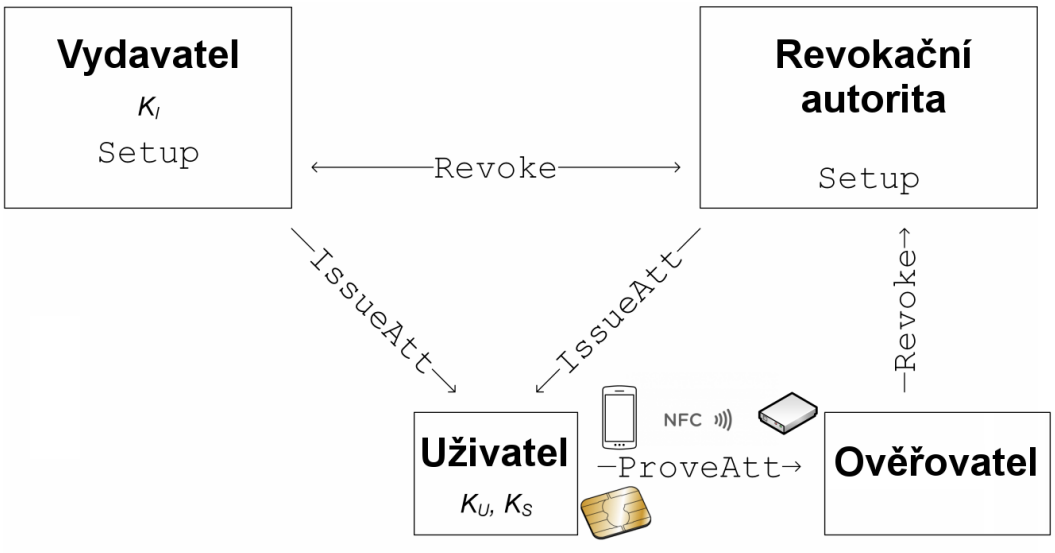
- **anonymita:** při ověřování zůstává identita uživatele skryta,
- **nespojitelnost:** jednotlivé relace ověření nelze přiřadit ke konkrétnímu uživateli,
- **nevysledovatelnost:** vydavatelé atributů nejsou schopni sledovat uživatele,

- **odhalení útočníka:** při porušení pravidel může být uživatel identifikován a vyřazen,
- **výběr atributů k odhalení:** uživatel může vybrat, které ze svých atributů odhalí k ověření,
- **odvolatelnost:** při ztrátě nebo vypršení platnosti se dají atributy odvolat.

Systém umožňuje uživateli, který je držitelem smart–karty, bezpečně prokázat své atributy u ověřovatele. Toto prokázání je zcela anonymní a nelze u něj identifikovat podobnost s jakýmkoli jiným předchozím prokázáním, tudíž nelze identifikovat stejného uživatele vícekrát.

Pro ochranu ověřovatele je zase v systému přítomna veřejná autorita, která dokáže odhalit škodlivé uživatele – útočníky.

Celé architektura protokolu HM12 je zobrazena na obrázku 2.1.



Obr. 2.1: Architektura systému HM12

2.2 Entity protokolu

Protokol obsahuje celkem čtyři entity. Jejich role jsou následující [2].

2.2.1 User

Uživatel (angl. User) je reprezentován smart-kartou s vydanými atributy od vydavatele. Jeho cílem je anonymní využívání služeb, bez možného ze sledování a zneužívání soukromých dat.

Každá čipová karta má svůj unikátní soukromý klíč K_U. Pro každé ověřování je poté vygenerován soukromý klíč K_S, díky kterému není možní spojit toto ověřování s předchozími relacemi ověření.

2.2.2 Issuer

Vydavatel (angl. Issuer) vydává osobní informace uživateli. Vydavatel zároveň spolupracuje s ověřovatelem a odvolávatelem v případě porušení pravidel a odhalování útočníka. Vydavatel vlastní soukromý klíč K_I.

2.2.3 Verifier

Cílem ověřovatele (angl. Verifier) je poskytování služeb pouze oprávněným klientům a identifikace útočníků. Při ověřování se zjistí, jestli daný uživatel vlastní konkrétní atribut. Může probíhat i offline, takže ověřovatel nemusí komunikovat s žádnou další entitou.

2.2.4 Revocation Referee

Z angličtiny se dá tato entita přeložit jako odvolávatel. Cílem odvolávatele je dohlížet na identifikaci útočníků. Rozhoduje o zrušení nespojitelnosti nebo anonymity uživatele na základě důkazů, které jsou poskytovány ověřovatelem. Aby toho byl odvolávatel schopen, spolupracuje s vydavatelem a ověřovatelem. Odvolávatel generuje parametry systému $params$ a vlastní soukromý klíč K_{RR}.

2.3 Protokoly

Schéma protokolu HM12 je rozděleno do čtyř protokolů: Setup, IssueAtt, ProveAtt a Revoke .

2.3.1 Setup

Tento protokol probíhá mezi odvolávatelem a vydavatelem. Jako vstup jsou použity bezpečnostní parametry (k, l, m) a jako výstup systémové parametry *params*, vydavatelův a odvolávatelův soukromý klíč.

Bezpečnostní parametr k udává délku funkce hash, l je délka uživatelova soukromého klíče a m je parametr chybovosti.

Systémové parametry params obsahují hodnoty q, p, h_1, h_2, n, g_1, g_2, g_3 a A_{seed} jsou veřejné. Parametr A_{seed} je identifikátor konkrétního atributu [2].

2.3.2 IssueAtt

Tento protokol vytváří uživatelův soukromý klíč K_U, který se používá pro budoucí ověření vlastněných atributů. Uživatel si tento klíč uloží na smart-kartu. Generovaný klíč je známý pouze uživateli a pro jeho odhalení musí spolupracovat vydavatel a odvolávatel.

První část protokolu probíhá mezi vydavatelem a uživatelem. V této části musí uživatel poskytnout vydavateli důkaz o svojí identitě a o tom, že vlastní dané atributy.

Druhá část protokolu poté probíhá mezi uživatelem a odvolávatelem. Uživatelův soukromý klíč se skládá z hodnot w_1, w_2 a w_{RR} [2].

2.3.3 ProveAtt

Úkolem ověřovacího protokolu je prokázat, že uživatel vlastní platné atributy a je oprávněn využívat služeb ověřovatele. K tomu protokol používá veřejné *params* a uživatelův soukromý klíč K_U uložený na smart–kartě [2]. Uživatelův klíč je však pokaždé modifikován tak, aby nemohlo dojít k vzájemnému spojení relací. V této fázi je tak uživatel zcela anonymní .

2.3.4 Revoke

Tento protokol se používá pro odhalení skutečné identity uživatele a vyřazení potencionálního útočníka ze systému. Pokud má ověřovatel důkazy o škodě, předá je odvolávateli. Ten může poté zvolit typ odvolání.

3 IMPLEMENTACE WEBOVÉ VIZUALIZACE PROTOKOLU HM12

V této kapitole je popsáno samotné řešení bakalářské práce. Cílem je vytvořit interaktivní webovou aplikaci pomocí jazyka HTML5 a Javascriptu, která bude demonstrovat funkčnost protokolu HM12. Toho je dosaženo nástrojem Adobe Edge Animate, kterým se vytvoří interaktivní webová stránka s animacemi. Poté je potřeba do něj implementovat funkce a výpočty, které provádí protokol HM12 na stranách svých entit. Výpočty jsou realizovány ve skriptech jazyka Javascript.

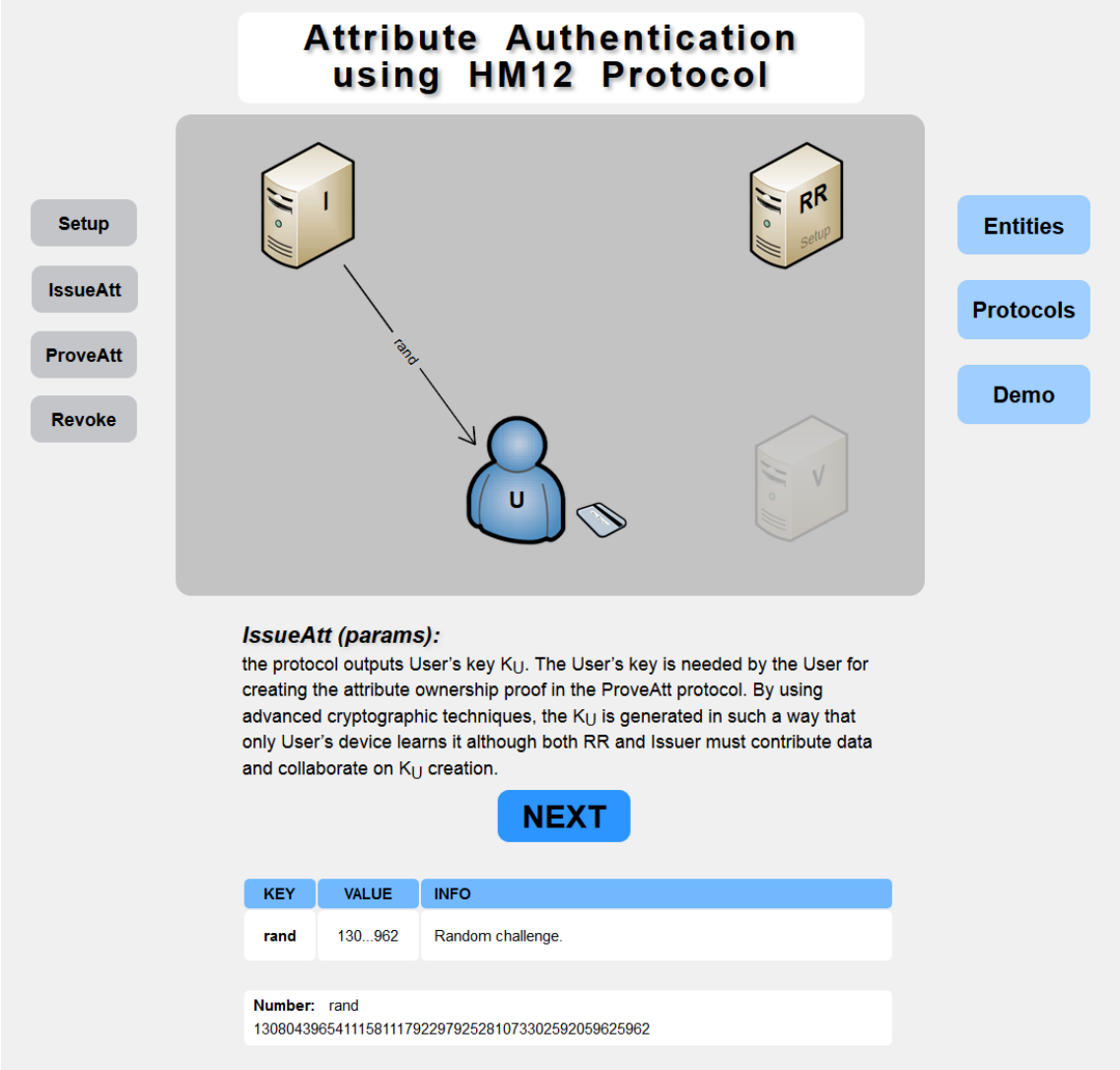
Webová stránka bude obsahovat základní popis entit a komunikačních protokolů. Dále by stránka měla být schopna zobrazit funkci protokolů a komunikaci mezi entitami včetně vypočítaných klíčů a parametrů. V poslední řadě bude graficky demonstrovat práci celého systému v sekci Demo.

3.1 Webová stránka

3.1.1 Uživatelské prostředí

V hlavním okně se na pravé straně nacházejí 3 tlačítka, které odkazují na 3 hlavní sekce. Jsou to zobrazení přehledu entit, protokolů a demo.

Sekce s přehledem entit a protokolů jsou podobné. V obou případech se zobrazí celé schéma protokolu HM12 a na levé straně se objeví menu s příslušnými tlačítky. V sekci s entitami to budou tlačítka se jmény entit a u protokolů zase s názvy jednotlivých protokolů. Po kliknutí na některé tlačítko z levého menu se zviditelní související část schématu, resp. se částečně zprůhlední zbytek schématu, a dole pod schématem se objeví popis daného protokolu či entity. V sekci protokolů je navíc k dispozici animovaný demonstrátor funkčnosti. Jeho úkolem je předvést komunikaci mezi entitami vybraného protokolu. Jednotlivé kroky komunikace lze spouštět tlačítkem Run. Funkce demonstrátoru vypadá tak, že se ve schématu systému objeví šipka, která vyjede z některé entity směrem k jiné entitě. Tato šipka s sebou nese seznam kryptografických klíčů a parametrů, které jsou v daném kroku mezi entitami předávány. V textovém poli pod schématem je vždy tabulka s popisem všech posílaných hodnot v daném kroku. Tabulka obsahuje název parametru, číselnou hodnotu a krátký orientační popis. Protože některá čísla mohou mít rozměr až 1024 bitů, je uváděn pouze zkrácený výpis, tj. tři první a tři poslední číslice. Pod tabulkou je proto k dispozici informační okno, které na několika řádcích zobrazí celé číslo, při najetí kurzoru na některé z nich. Vzhled webové stránky a sekce s protokoly je vyobrazen na obrázku 3.1.



Obr. 3.1: Náhled webové stránky - protokol ProveAtt

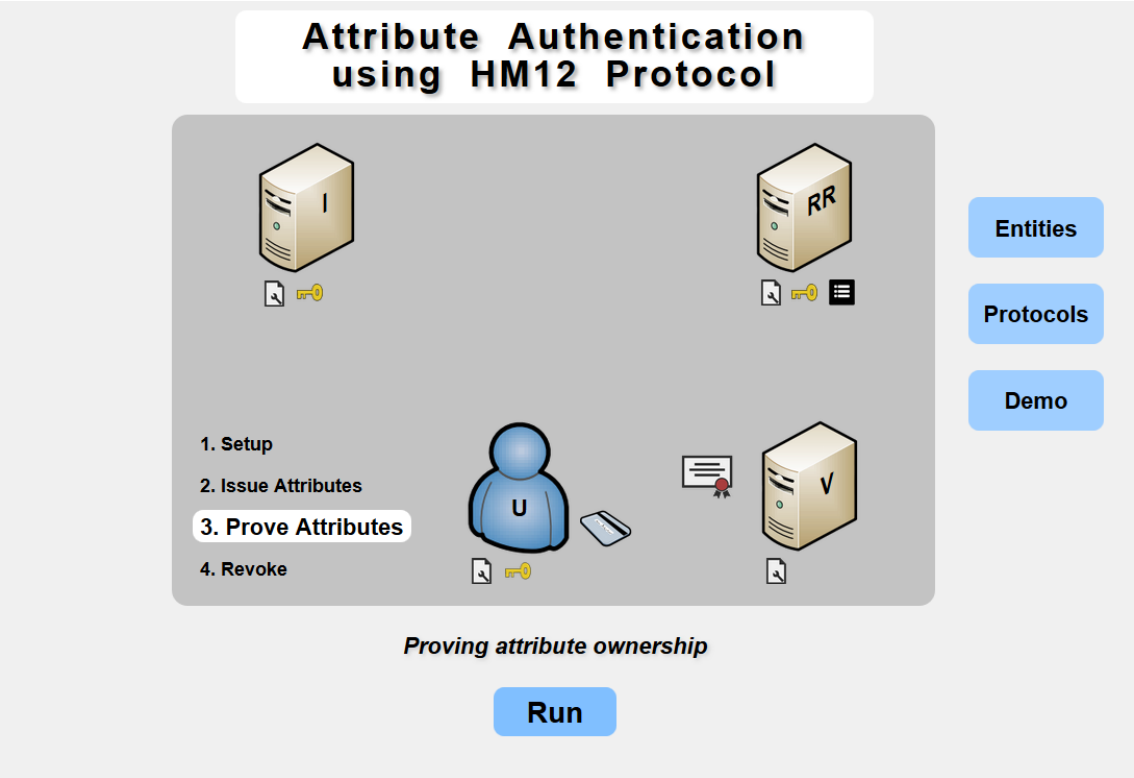
Na základě tohoto demonstrátoru lze nahlédnout na hodnoty systémových klíčů, parametru a jiných proměnných v jednotlivých krocích. Při obnovení stránky se výpočty protokolů provedou znovu, proto budou číselné hodnoty vždy jiné. To je také důkazem, že jsou ve webové stránce kryptografické výpočty opravdu implementovány a nejde pouze o statická čísla.

Sekce demo demonstruje celou funkčnost systému v jednotlivých krocích. V této sekci se na webové stránce (obr. 3.2) objeví schéma se zvýrazněnými entitami a pod schématem tlačítko Run/Pause. Demonstrace je pojata jako animovaný sled událostí a skládá se z kroků:

1. nastavení systému,
2. generování a vydání uživatelského klíče,
3. ověřování uživatele,

4. vyřazení uživatele ze systému.

Animaci je možné pozastavit a zase spustit pomocí výše zmíněného tlačítka Run/Pause. Animace zobrazuje komunikační kroky mezi entitami prostřednictvím pohyblivých obrázkových ikon, které reprezentují systémové parametry nebo klíče. Jde vlastně o spojení animací jednotlivých protokolů ze sekce Protocols do jedné celistvé animace, která je navíc zjednodušena použitím ikon místo výčtu parametrů a zobecněním jejich postupů.



Obr. 3.2: Náhled webové stránky - demo

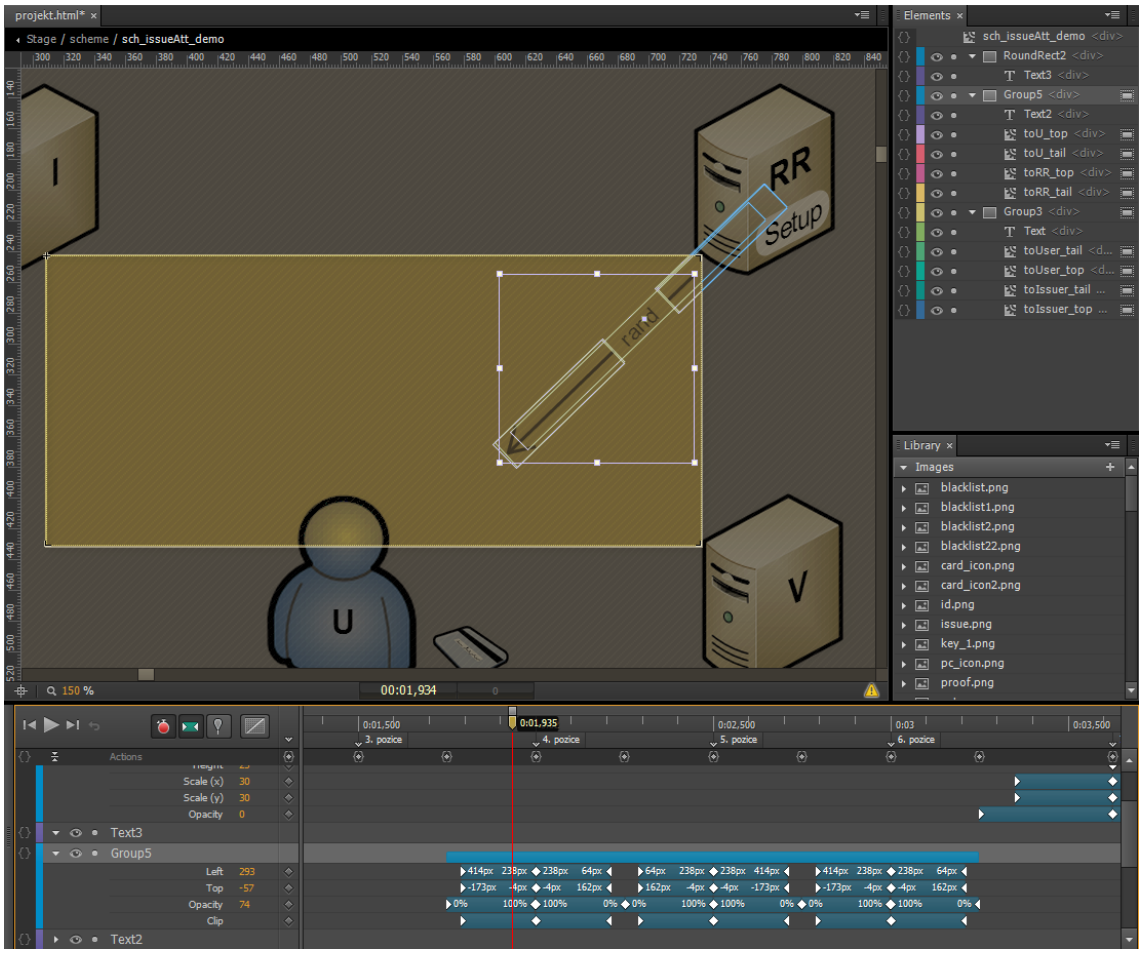
3.1.2 Konstrukce webu a animací

Pro správné fungování webové aplikace je na základě požadavků zadání potřeba dopředu připravit vhodné rozvržení celé stránky. Jedná se o konstrukci stránky, tzn. z jakých animací se bude skládat, jaké budou mít strukturu a jak se budou ovládat. Na těchto kritériích pak závisí výsledná složitost, přehlednost, rychlost a spolehlivost aplikace.

V tomto případě webová stránka tvoří hlavní plátno animací, tzv. `Stage`. Na plátně pak probíhá veškerá prezentace dat pomocí vytvořených symbolů a jejich dílčích animací. Použité symboly se dají rozdělit na jednoduché a hlavní. Ty jednoduché obsahují nějaký statický objekt, např. text, popisek nebo tlačítko s definovanou

animací, obvykle zviditelnění, zneviditelnění, případně pohyb (vstup na plátno). Za hlavní animační symbol se dá považovat symbol `scheme`, který obsahuje řadu dalších jednoduchých symbolů např. ikony entit, šipky protokolů nebo symboly s animací demonstrací. Výhoda takového zapouzdření spočívá jednoduché práci s prvky jako s celkem. Jednotlivé symboly prvků schématu se pak spouštějí samostatně podle toho, v které části aplikace se uživatel nachází. Spouštěním animací se podrobněji věnuje kapitola 3.1.3. Samotný symbol `scheme` tak animuje pouze zviditelňování dílčích symbolů na základě jejich vyvolávání tak, že animace přeskočí na úsek časové osy, který je vyhrazen pro onen konkrétní prvek. Díky časovému členění a skákání na časové ose jsou animace nezávislé na pořadí a schéma se může měnit libovolně.

Jako příklad tvorby animací je zde popsán symbol `sch_issueAtt_demo` (obsažený v symbolu `scheme`), který představuje demonstraci protokolu IssueAtt. Celý symbol je tvořen HTML oddílem `<div>`, který obsahuje dílčí prvky (obr. 3.3). Tyto prvky jsou pak vyobrazeny na časové ose a právě zde je vytvořena celá animace demo. V různých časech na časové ose jsou u každého objektu přidávány tzv. keyframes neboli klíčové snímky. Každý má nastavené různé vlastnosti daného objektu, v tomto případě pozici, rozměry, okraje a průhlednost. Propojením dvou keyframes dojde mezi nimi k vytvoření animovaného přechodu. Tímto způsobem se na časové ose vytvoří všechny potřebné sekvence. Kromě keyframes se na časovou osu přidávají také události. Jde o část javascriptového kódu, který se spustí v daný okamžik. Pomocí těchto načasovaných událostí je řízen běh (zastavování) animace a hlavně spouštění animací jiných symbolů, např. zobrazování textu pod schématem.



Obr. 3.3: Vytváření animace protokolu IssueAtt

3.1.3 Řetězové spouštění animací a řídící proměnné

Celá aplikace musí pro správnou funkčnost dodržovat správný sled animací. Situace je navíc komplikována faktem, že jednotlivé animace jsou nezávislé a jejich sled se neustále mění podle chování návštěvníka stránek. Kvůli tomu je třeba použít zvláštní systém spouštění a řízení, který musí hlídat spoustu zvláštních situací.

Ve JS skriptech jsou za tímto účelem vytvořeny řídící proměnné (obr. 3.4), které ukládají aktuální pozici návštěvníka nebo vyobrazenou sekci. Při každé akci uživatele (stisknutí jakéhokoli tlačítka) se vykoná série algoritmů, které detekují scénu, postupně spustí odchodovou animaci aktuální scény a zobrazí vstupní animaci nové scény. Problém ale nastává v situacích, že některé části schématu jsou společné, nebo jistým způsobem přechází do scény nové. V těchto nestandardních situacích algoritmy na základě detekce aktualizují pouze potřebnou část obsahu.

```
11  var scena = "text-1";
12  var btn_menu = "entities_btnMenu";
13  var demo_status = "stop";
14  //var scheme_items = new Array ("sch_Issuer","sch_IssueAtt","sch_ProveAtt","sch_Revol
15  var scheme_item = "";
16  var i = 0;
```

Obr. 3.4: Inicializace řídících proměnných

Řetězové spouštění animací se využívá převážně v sekci demo, jak již nastínila kapitola 3.1.2. Princip spočívá ve spuštění jedné animace, která postupně v různých časech spouští animace jiných symbolů nebo vykonává různé akce potřebné ke správnému zobrazení scény. Za tímto účelem jsou v časové ose rozmístěny javascriptové časované triggery (spouštěče). Například symbol `scheme_demo` postupně zobrazuje pohybující se ikony parametrů. Nejprve však spustí animaci zneviditelnění části schématu, která se nachází v symbolu o úroveň výš. Poté v průběhu animace spouští zvýraznění aktuálního kroku v levém seznamu. Na závěr animace spustí své zneviditelnění a schéma uvede do původního stavu.

3.2 Práce v Javascriptu

3.2.1 Použité knihovny

Aby bylo možné realizovat a prezentovat výpočty, které demonstrovaný protokol HM12 provádí, je nutné do webové stránky naimportovat několik skriptů. Do `html` souboru je tak vložen jeden hlavní skript `keys_init.js`, který ve svém kódu provádí inicializaci proměnných potřebných k výpisu kryptografických klíčů a parametrů na web. Obsahuje také výpočetní část, kde se pomocí web workerů spouští v paralelním vlákně skript `vypocet_hm12.js` obsahující samotnou implementaci výpočtů protokolu HM12. Výpočty protokolu HM12 a použití web workerů je rozebráno v následujících kapitolách 3.2.3 a 3.2.2.

Abychom vůbec mohli realizovat výpočty protokolu HM12, je nutné vyřešit největší problém – datový typ. Jelikož datové typy v Javasriptu neumožňují zápis čísla o velikosti 1024 bitů, je potřeba vytvořit knihovnu a takový datový typ definovat. V ostatních jazycích se pro takové čísla používá datový typ BigInteger (např. Java). Po vzoru těchto programovacích jazyků byl vytvořen datový typ BigInteger i pro Javascript, avšak pouze jako nadstavba. Tento datový typ je definovaný v externích knihovnách, které jsou volně dostupné na internetu, stačí je pouze importovat do kódu `html` souboru a používat jej s dodržením požadovaných syntaxí [10].

Ve výpočtech se dále využívá hashovací funkce SHA1. Tato funkce vytvoří ze vstupních hodnot 160 bitový otisk (hash), typicky hexadecimální číslo. Funkce SHA1

je v Javascriptu již vytvořena a volně přístupná na internetu. Stačí ji proto do kódu opět pouze importovat a funkci poté zavolat.

Vložené skripty pak vypadají následovně:

```
<script src="keys_init.js" type="text/javascript"></script>
```

v `html` souboru webové stránky a

```
importScripts('http://crypto-js.googlecode.com/svn/tags/3.1.2/build
/rollups/sha1.js');
importScripts('bigInteger.js');
```

pro importování knihoven přímo ve skriptech.

3.2.2 Multithreading a práce na pozadí

Jednou z největších nevýhod JavaScriptu je, že současné implementace provádí všechny skripty pouze v jednom vlákně. Pokud je někde vytvořen příliš složitý výpočet (nebo kód obsahuje chybu, která vyústí v zacyklení), přestane web reagovat a nezpracovává události vyvolané uživatelem. V horších případech může dojít k zaseknutí celého prohlížeče, který zároveň využívá 98% dostupného výkonu CPU. Udělat JavaScript vícevláknový by znamenalo udělat jej znovu (a jinak), proto přichází Web Workers s řešením, které za jistých okolností umožní spuštění více vláken najednou [4]. Web Workers se zpravidla využívají pro zpracování výpočetně a časově náročných skriptů, které by jinak blokovaly uživatelské rozhraní. Nicméně používání Web Workers s sebou nese několik komplikací. Web Workers nemají přístup k obsahu stránky, takže je nutné naprogramovat komunikaci mezi vláknem uživatelského rozhraní a pracovním vláknem. Tato komunikace probíhá prostřednictvím zpráv (událostí `message`) [6].

V této práci je pomocí Web Workers realizován výpočet všech částí kryptografického protokolu HM12. Jejich výpočet totiž může trvat v rozmezí 4 – 15 sekund v závislosti na výkonu hardwaru. V klasickém jednovláknovém zpracování skriptů by web po celou tuto dobu nereagoval a ani se nezobrazil.

Ve hlavním skriptu `keys_init.js` proběhne inicializace proměnných, potřebných pro uložení výsledků výpočtu protokolu. Poté dojde k vytvoření objektu `Worker`, který odkazuje skript s výpočty. Tím se spustí Web Worker na paralelním vlákně. Po dokončení všech výpočtů Worker pošle zprávu rodičovskému skriptu a předá mu pole výsledných dat prostřednictvím zprávy `postMessage()`. Tím dojde v rodičovském skriptu k obsluze události `onessage` a následnému uložení přijatých dat. Výsledky se poté okamžitě zobrazí na webové stránce. Zdrojový kód rodičovského skriptu pak vypadá následovně:

```
var worker = new Worker('vypocet_hm12.js');
worker.onmessage = function (event) {
  w1_issue1 = event.data[0];
  w2_issue1 = event.data[1];
  ...
};
```

3.2.3 Kryptografické výpočty

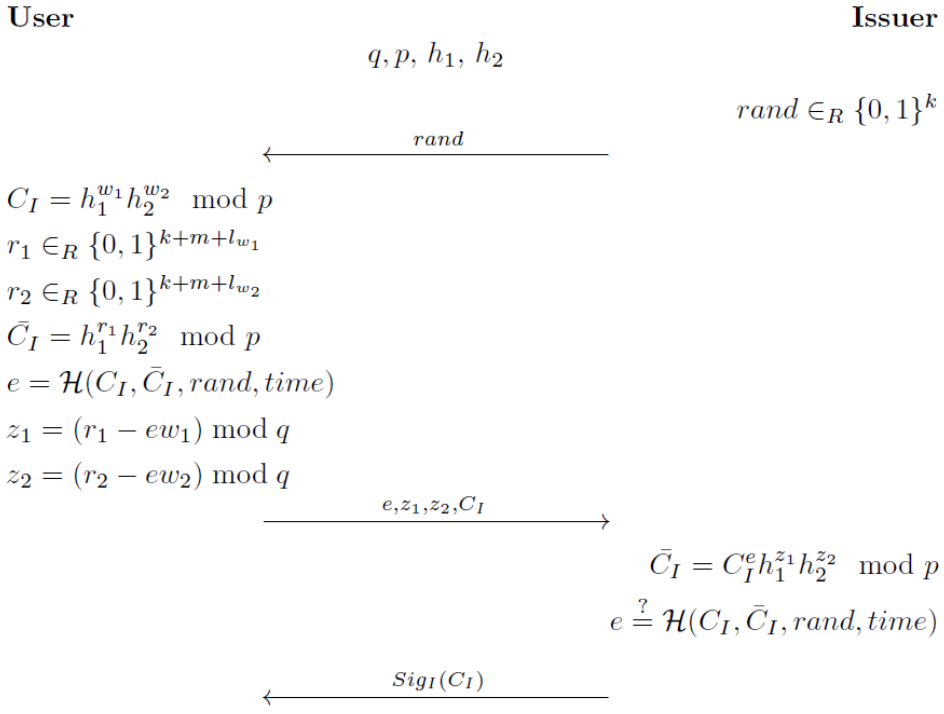
Protokol Setup

Protokol Setup má za úkol vygenerovat systémové parametry *params*, odvolávatelův revokační klíč K_{RR} a vydavatelův klíč K_I. Setup nejprve přijme systémové bezpečnostní parametry k, l, m. Poté již vygeneruje skupinu parametrů a klíčů podle předepsaných matematických operací. Nakonec odovolávatel vypočítá identifikátor atributu A_{seed}, který bude odkazovat na konkrétní informaci o uživateli [2]. Těchto identifikátorů může existovat více typů, podle toho, jaký atribut chce uživatel ověřit.

Hodnoty q, p, h_1, h_2, n, g_1, g_2, g_3 a A_{seed} se zveřejní pro všechny entity jako systémové parametry *params*, zatímco r, s, S_1, S_2 a S_3 zůstanou uloženy u odvolávatele jako klíč K_{RR}.

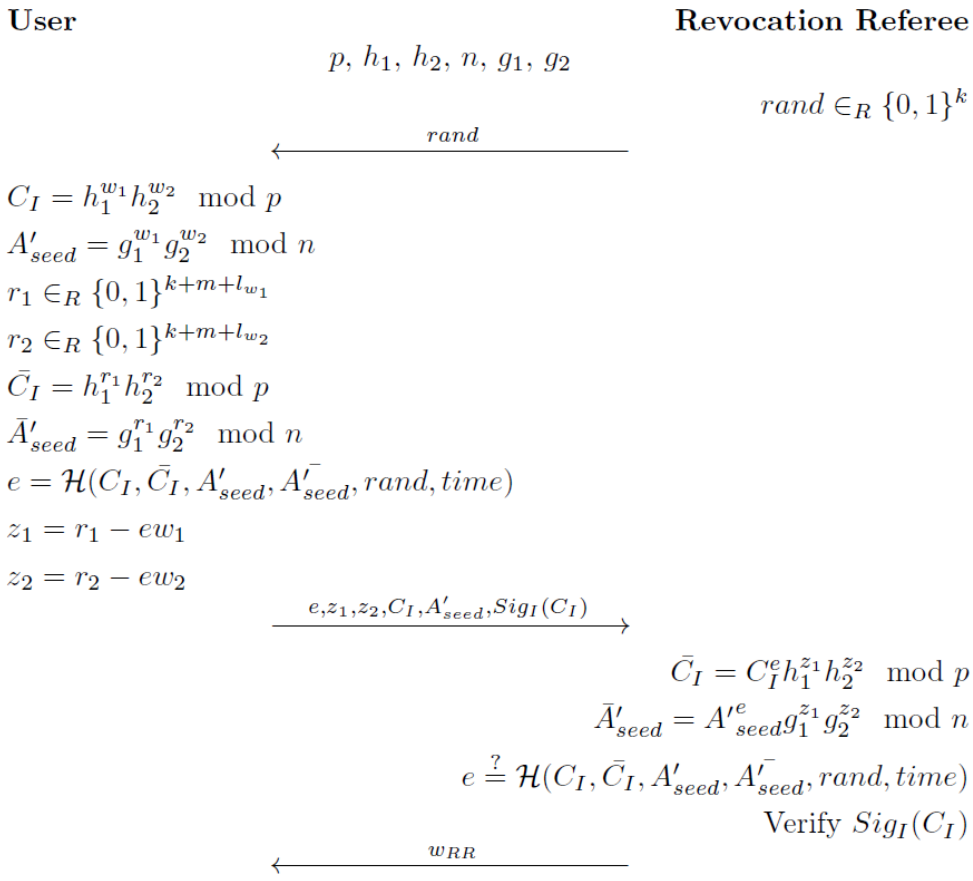
Protokol IssueAtt

Protokol IssueAtt je rozdělen na dvě části. V první části probíhá komunikace mezi uživatelem a vydavatelem [2]. Uživatel vygeneruje hodnoty w_1 a w_2, jenž představují části jeho uživatelského klíče K_U. Vygeneruje se také hodnota C_I, která je digitálně podepsána. Pomocí hashovací funkce SHA-1 se vytvoří otisk, který se spolu s dalšími hodnotami posílá vydavateli. Vydavatel ověří podpis a nového uživatele potvrdí zasláním svého digitálního podpisu.



Obr. 3.5: Schéma protokolu IssueAtt – část 1

V druhé části protokolu IssueAtt komunikuje uživatel s odvolávatelem. Opět zde probíhá výpočet hodnot potřebných pro přenos údajů. Po přijetí parametrů odvolávatel zkontroluje podpis vydavatele a následně vypočítá hodnotu w_{RR}, kterou uživatel potřebuje pro svůj soukromý uživatelský klíč K_U [2]. Ten představuje výsledná trojce hodnot w_1, w_2 a w_{RR}. Klíč K_U bude v budoucnu sloužit pro ověřování uživatelových atributů.



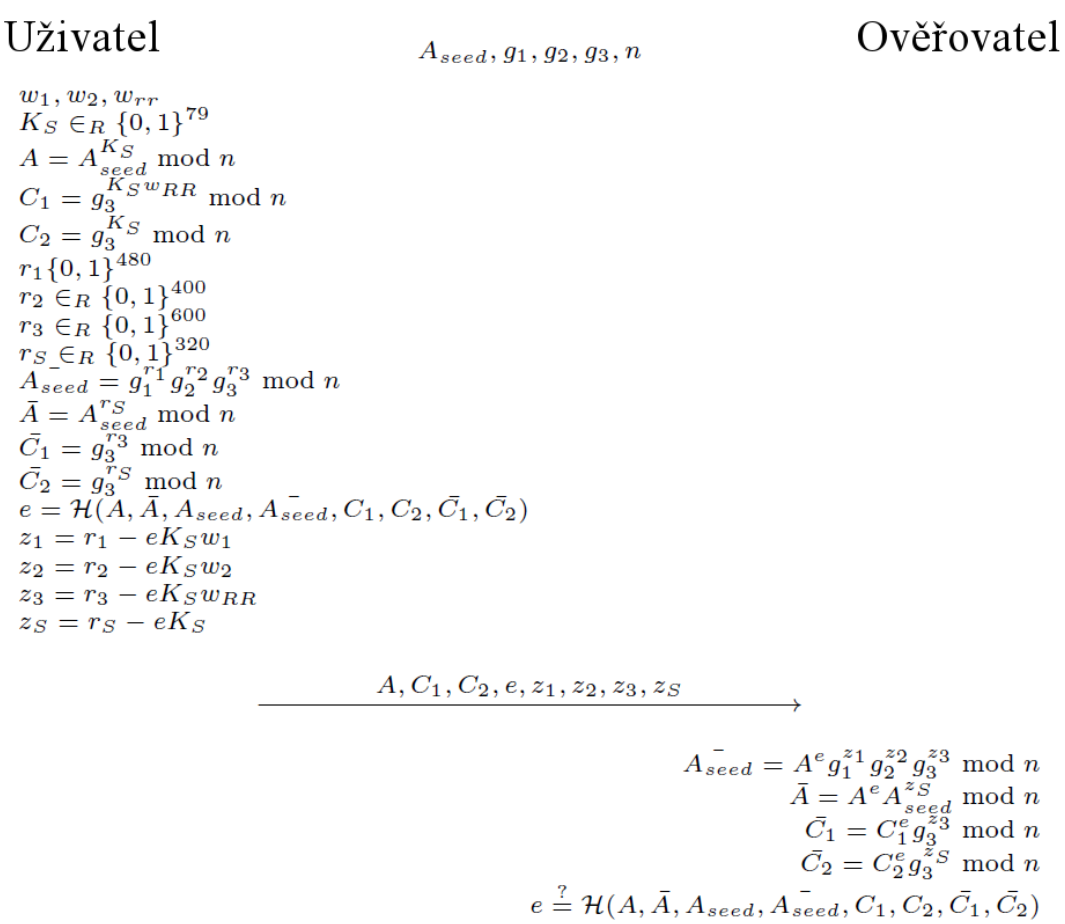
Obr. 3.6: Schéma protokolu IssueAtt – část 2

Protokol ProveAtt

Tento protokol nejprve na straně uživatele obdrží veřejné systémové parametry $(A_{seed}, g_1, g_2, g_3, n)$ a soukromý uživatelský klíč (w_1, w_2, w_{RR}) a provede se výpočet protokolu. Klíč, který dodává odvolávatel je potřeba k ochraně před předkládáním falešných klíčů, které by mohli umožnit odhalení citlivých dat.

První se na straně uživatele vygeneruje soukromý klíč K_S jako náhodné číslo, jenž znáhodňuje přenos a zajišťuje nespojitelnost více relací s daným uživatelem. Na základě získaných klíčů protokol vypočítá další parametry, které zpracuje hashovací funkce a vytvoří otisk. Ten se poté s některými parametry posílá na stranu ověřovatele, který z přijatých a veřejných parametrů zpětně vypočítá požadované klíče. Porovnání těchto hodnot na obou stranách pak vede k vyhodnocení. Výsledkem protokolu jsou dvě možnosti, uživatel je buď schválen nebo zamítnut.

Podrobný popis matematických operací v protokolu je znázorněn na obrázku 3.7.



$K_S \in_R \{0,1\}^l$ - náhodně generované číslo K_S o délce l bitů
$\mathcal{H}$ - hash SHA-1 s výstupem 160 bitů
$A_{seed}, g_1, g_2, g_3, n, w_1, w_2, w_{rr}$ - zadáno

Obr. 3.7: Schéma protokolu ProveAtt

Protokol Revoke

Tento protokol se vykonává v případě porušení pravidel systému, kdy je potřeba vyřadit ze systému konkrétního uživatele nebo ho identifikovat. O typu odvolání rozhoduje odvolávatel [2].

Ověřovatel pošle odvolávateli klíče C_1 a C_2, díky kterým odvolávatel může určit relační klíč K_S a část uživatelského klíče – w_{RR}. Odvolávatel přidá informaci o revokaci na veřejný blacklist uživatelů rev, který se pošle zpět ověřovateli. Na základe tohoto blacklistu může každý ověřovatel v systému blokovat škodlivého uživatele. V tomto případě nedochází k odhalení identity uživatele, pouze je vyřazen ze systému.

Když je potřeba škodlivého uživatele identifikovat, odvolávatel najde v databázi příslušný klíč C_I, který byl spolu s w_{RR} dodán v rámci protokolu IssueAtt. C_I je přeposlán vydavateli, který najde ve své databázi přislušný digitálně podepsaný klíč C_I. Na základě těchto znalostí je možné konkrétního uživatele identifikovat. Odvolávateli jsou poté přeposlány informace o jeho identita.

Aby bylo možné realizovat výpočty těchto protokolů v Javascriptu, bylo nutné naprogramovat navíc několik matematických funkcí, které neposkytuje použitá nadstavbová knihovna pro datový typ BigInteger. Ve většině případů jsou funkce psány obecně pro běžný datový typ, jen se nakonec pozmění syntaxe příkazů pro datový typ BigInteger [10]. V některých funkcích však bylo nutné konstrukci navrhnout zcela na míru tomuto speciálnímu datovému typu.

V prvé řadě bylo nutné napsat algoritmus pro generování náhodného čísla v požadovaném rozsahu. Tento algoritmus využívá Javasriptové funkce `rand`, avšak ta má jen omezené maximální číslo. Proto je takto funkce volána několikrát a výsledek se pokaždé posouvá násobením na čím dál tím vyšší řád. Získaná čísla se pak sčítají již do datového typu BigInteger, dokud výstupní číslo nedosáhne požadovaného rozměru.

Protokol `Setup` při výpočtu využívá funkce pro hledání nejvyššího společného dělitele dvou čísel. Tato funkce využívá klasické dělení se zbytkem, jenž je v knihovně BigInteger definované, takže návrh nebyl složitý.

Funkce pro zjištění rozměru čísla v bitech (`bitSize`) využívá protkol IssueAtt. Dochází zde k převodu čísla na text. Rozměr lze poté zjistit klasickou JS funkcí `.lenght`.

Jako poslední bylo nutné vytvořit funkci inverzního modulárního mocnění pro datový typ BigInteger. Obecná matematická konstrukce tohoto algoritmu je rozepsána na primitivnější operace, které již bylo možné vykonat použitím knihovny BigInteger. Zdrojový kód této funkce je zde uveden jako příklad implementace v Javascriptu.

```
function inverseMod2(a,n)
{
  i = BigInteger(n);
  v = BigInteger(0);
  d = BigInteger(1);
  while (BigInteger(a).compare(0) > 0)
  {
    t = BigInteger(i).divide(a);
    x = BigInteger(a);
    a = BigInteger(i).remainder(x);
    i = BigInteger(x);
    x = BigInteger(d);
    d = BigInteger(v).subtract(BigInteger(t).multiply(x));
    v = BigInteger(x);
  }
  v = BigInteger(v).remainder(n);
  if (BigInteger(v).compare(0) < 0)
  v = BigInteger(BigInteger(v).add(n)).remainder(n);
  return v;
}
```

4 SPUŠTĚNÍ WEBOVÝCH STRÁNEK

Nakonec je potřeba výsledný webový demonstrátor zveřejnit a zpřístupnit jako klasickou webovou stránku. K tomu je potřeba mít nainstalovaný webový server. Na serverech se většinou používají operační systémy Linux a jeho různé distribuce, proto se uvedený příklad vztahuje k distribuci Debian. Nicméně instalace webového serveru je velmi jednoduchá a pro všechny distribuce téměř totožná. Instalace plnohodnotného webového serveru se skládá z instalace balíčků Apache, PHP a MySQL [8].

Apache HTTP Server je pravděpodobně nejrozšířenější webový server, jehož instalační balíček bývá obsažen ve většině linuxových distribucích. V Debianu jde o balíček `apache2`. Stačí jej proto pouze nainstalovat příkazem [1]:

```
aptitude install apache2
```

Po instalaci dojde ke spuštění serveru. Server obsahuje kontrolní `html` stránku, která se zobrazí ve webovém prohlížeči (do prohlížeče je nutné zadat IP adresu serveru). Měl by se objevit známý nápis „It works!“ značící, že je všechno v pořádku a webový server běží. Apache má soubory webové prezentace uložené v adresáři `/var/www`. Do tohoto adresáře je tedy nutné nakopírovat vytvořený webový demonstrátor protokolu HM12. Nyní se již demonstrátor nachází na webovém serveru a je přístupný z internetu jako klasická webová stránka.

K serveru Apache je však vhodné doinstalovat několik dalších balíčků, především balíček s podporou skriptovacího jazyka PHP. To se provede příkazem:

```
aptitude install libapache2-mod-php5 php5
```

V tuto chvíli je nutné restarovat Apache.

```
/etc/init.d/apache2 restart
```

Teď už by mělo být vše funkční. Pro otestování je nutné v adresáři `/var/www` vytvořit soubor např. `index.php` s obsahem:

```
<?php phpinfo(); ?>
```

Měla by se zobrazit detailní konfigurace PHP. Soubor pojmenovaný `index` spouští webový server defaultně při vstupu do složky webové prezentace. Aby se ale zobrazil webový demonstrátor, tak v se obsah tohoto souboru musí změnit na:

```
<?php include "projekt.html"; ?>
```

K webovému serveru Apache je možné doinstalovat celou řadu dalších balíčků, např. databázový systém MySQL, SSL, atd. Nicméně pro funkčnost webového demostrátoru protokolu HM12 již nejsou potřeba.

5 VÝSLEDKY PRÁCE

Výsledná stránka obsahuje okno – animační plátno, do kterého jsou postupně vkládány symboly s kontextovými nabídkami, obrázky či textovými poli. U každého symbolu se potom definují animace objektů. Na časové ose jsou vytvořeny intervaly jednotlivých animací, které jsou uvozeny volací značkou. Akce uživatele způsobují spouštění jednotlivých animací v konkrétních časech, což má za následek dynamické zobrazování požadovaného obsahu.

Nástrojem Adobe Edge Animate je vytvořena webová stránka spolu se skripty, definující její chování. Tyto Javascriptové funkce, které řídí chod aplikace a dynamické zobrazování, jsou navíc upravovány ručně znalostmi programátora.

Při spuštění webové stránky se jednotlivé skripty rozdělí do dvou procesorových vláken. V jednom se spustí a zobrazí přednastavená webová stránka a v druhém vlákně probíhají výpočty protokolu HM12. Protože jde o výpočty náročných operací s čísly, které mají rozměr až 1024 bitů, uživatel může postřehnout absenci kryptografických klíčů na již zobrazené webové stránce. Prakticky záleží na výkonu počítače, jak rychle dokáže tyto výpočty vykonat. Po výpočtu se na webové stránce všechny hodnoty zobrazí. Tyto výpočty byly podrobně popsány v kapitole 3.2.3. Díky rozdělení na více vláken však již nedochází ke zpozdění zobrazení webu a web reaguje na chování uživatele.

Spočítané kryptografické klíče se zobrazují v sekci Protocols v jednotlivých krocích demonstrace každého protokolu. Kvůli rozměrům čísel ale není možné přehledně zobrazit všechny klíče. Je nutné rozdělit čísla na zkrácený výpis (první tři a poslední tři číslice) a celý výpis. Celý výpis je dostupný v dodatečném informačním textovém poli po najetí kurzoru na některý ze zkrácených výpisu.

Výsledek práce představuje vytvořená webová stránka propojená s výpočty protokolu HM12 uloženého ve skriptech. Stránka obsahuje základní popis entit, protokolů a jejich vypočtených klíčů. Rovněž stránka obsahuje animovaný demonstrátor funkčnosti celého systému, kde je ve spojité animaci co nejjednodušeji prezentováno postupné vykonávání protokolů. Všechny sekce webové stránky jsou propojeny sérií přechodových animací tak, aby v uživateli budily co nejlepší dojem a stránka se jevila jako pokročilá aplikace. Za tímto účelem bylo nutné zavést systém pro správné přepínání a spouštění animací.

6 ZÁVĚR

Cílem této práce je navrhnout a vytvořit interaktivní webové stránky s demonstrátorem kryptografického protokolu HM12.

V teoretické části práci byla provedena analýza v současné době nejpoužívanějších webových technologií, prioritně nové specifikace jazyka HTML – HTML5, obecného principu tvorby animování, analýza animačního nástroje, dále problematiky současné kryptografie a analýza kryptografického protokolu HM12.

Při zkoumání možností dostupných webových technologií vyplynulo, že pro tuto práci bude nejvhodnější využít vlastností HTML5. Především z důvodu jednodušší implementace animací, které podporuje samotný jazyk a není tak potřeba vkládat další technologie jako Flash, nebo Javu. Matematickou část práce lze realizovat skriptovacím jazykem Javascript, který je nedílnou součástí dynamických webových stránek, tudíž i HTML5 animací.

Praktická část práce představuje vytvoření samotného webového prostředí použitím výše zmíněné technologie a implementace funkčního protokolu HM12. Webové prostředí a animace je vytvořeno pomocí nástroje Adobe Edge Animate.

V další fázi praktické části práce je realizován výpočet samotného protokolu HM12. Ten se skládá z několika dílčích protokolů, které na sebe postupně navazují.

Výpočty protokolu HM12 provádí samostatný skript vložený do hlavní webové stránky, avšak kvůli časové náročnosti jsou zpracovány v paralelním procesorovém vlákně. Kryptografická primitiva jsou vytvořena podle matematické předlohy dodané vedoucím práce.

Závěrem lze konstatovat, že vytvořená webová stránka splňuje stanovené požadavky a svůj účel. Bylo vytvořeno prostředí pro realizaci a prezentaci funkčního modelu kryptografického systému v prostředí webového prohlížeče. Výsledná webová stránka dokáže zpracovat veškeré výpočty kryptografického protokolu HM12 a zároveň vysvětluje jeho funkci pomocí animovaných demonstrací.

LITERATURA

[1] DOČEKAL, M. Instalace LAMP. *Správa linuxového serveru* [online]. 2010 1(36) [cit. 2015-05-17]. Dostupné z URL: <http://www.linuxexpres.cz/praxe/sprava-linuxoveho-serveru-instalace-lamp>.

[2] HAJNÝ, J.; MALINA, L. *Unlinkable Attribute-Based Credentials with Practical Revocation on Smart-Cards.* Brno, 2012. Dostupné z URL: <http://cardis.iaik.tugraz.at/proceedings/cardis_2012/CARDIS2012_5.pdf>.

[3] LEHKÝ, P. *Technologie tvorby webových aplikací* [online]. Brno, 2007 [cit. 2014-12-11]. Dostupné z URL: <http://is.muni.cz/th/4246/fi_m/diplomova_prace-technologie_tvorby_webovych_aplikci.pdf>. Diplomová práce. MASARYKOVA UNIVERSITA FAKULTA INFORMATIKY.

[4] MALÝ, M. Multithreading s WebWorkers. *Webdesignérův průvodce po HTML5* [online]., 2010 1(31) [cit. 2015-05-17]. Dostupné z URL: <http://www.zdrojak.cz/clanky/webdesigneruv-pruvodce-po-html5-multithreading-s-webworkers/>.

[5] ORGOŇ, M. *Pokročilé bezpečnostní aplikace pro Android* [online]. Brno, 2014 [cit. 2014-12-11]. Dostupné z URL: <https://www.vutbr.cz/www_base/zav_prace_soubor_verejne.php?file_id=85089>. Diplomová práce. VUT Brno. Vedoucí práce Ing. JAN HAJNÝ, Ph.D.

[6] SALVET, P. Web Workers: JavaScript s více vlákny. *Interval.cz* [online]., 2011, (1) [cit. 2015-05-19]. Dostupné z URL: <https://www.interval.cz/clanky/web-workers-javascript-s-vice-vlakny/>.

[7] ŠŤASTNÝ J., LEHOCKÝ Z. HTML5 – nové vlastnosti. *Programujte.com* [online]. 11. 7. 2011 [cit. 2015-05-19]. Dostupné z URL: <http://programujte.com/clanek/2010082200-html5-nove-vlastnosti/>.

[8] ZMEK, T. Instalace Apache, PHP5, MySQL – Ubuntu. In: *ZMEK: Weby, domény, servery, aplikace* [online]. 2015 [cit. 2015-05-19]. Dostupné z URL: <http://zmek.eu/blog/instalace-apache-php5-mysql-ubuntu>.

[9] Adobe Edge Animate. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001 [cit. 2014-12-11]. Dostupné z URL: <http://en.wikipedia.org/wiki/Adobe_Edge_Animate>.

[10] *Biginteger.js* [online]. [cit. 2015-05-19]. Dostupné z URL: <http://john-edwin-tobey.org/Scheme/javascript-bignum/docs/files/biginteger-js.html>.

[11] HTML5. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001 [cit. 2014-12-11]. Dostupné z URL: <http://cs.wikipedia.org/wiki/HTML5>.

[12] Javascript. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001 [cit. 2014-12-11]. Dostupné z URL: <http://cs.wikipedia.org/wiki/JavaScript>.

[13] Javacsript – návody na použití jazyka.*Jak psát web.* [online]. [cit. 2015-05-19]. Dostupné z URL: <http://www.jakpsatweb.cz/javascript/>.

[14] Webová stránka. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001 [cit. 2014-12-11]. Dostupné z URL: <http://cs.wikipedia.org/wiki/Webova_stranka>.

SEZNAM PŘÍLOH

A OBSAH ELEKTRONICKÉ PŘÍLOHY

Přiložené CD obsahuje tyto soubory a složky:

- **edge_includes** – složka se systémovými skripty aplikace Adobe Edge Animate;
- **images** – složka s obrázky, vloženými do webové stránky;
- **symobly** – složka s exportovanými a importovanými animačními symboly;
- **biginteger.js** – knihovna s datovými typem BigInteger pro Javascript;
- **keys_init.js** – skript pro inicializaci proměnných potřebných k výpisu na web;
- **pocitani_bigint.js** – skript s výpočtem protokolu HM12;
- **projekt.html** – výsledná webová stránka
- **projekt.an** – soubor představující celý projekt v programu Adobe Edge Animate;
- **projekt_edge.js** – skript s nastaveném vlastností celé webové stránky a jednotlivých objektů;
- **projekt_edgeActions.js** – skript obsahující informace o animacích;
- **projekt_edgePreload.js** – nastavení projektu programu Adobe Edge Animate;
- **xsikor10_BakalarskaPrace.pdf** – elektronická verze této práce;

Webová stránka se spouští otevřením souboru **projekt.html** v internetovém prohlížeči. Soubory webové stránky však musí být uloženy na běžícím webovém serveru, který může být i lokální (localhost).