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

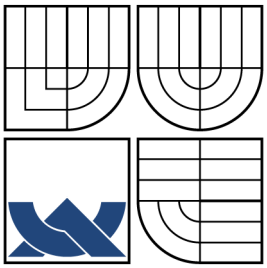
SOFTWAROVÁ PODPORA VÝUKY KRYPTOGRAFICKÝCH PROTOKOLŮ

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

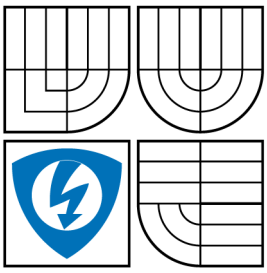
AUTOR PRÁCE
AUTHOR

BC. TOMÁŠ MAREK

BRNO 2009



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

SOFTWAROVÁ PODPORA VÝUKY KRYPTOGRAFICKÝCH PROTOKOLŮ

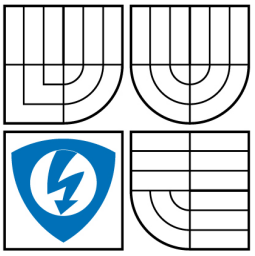
SOFTWARE SUPPORT OF TEACHING OF CRYPTOGRAPHY PROTOCOLS

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — BC. TOMÁŠ MAREK
AUTHOR

VEDOUCÍ PRÁCE — ING. KAREL BURDA CSC.
SUPERVISOR

BRNO 2009



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Diplomová práce

magisterský navazující studijní obor
Telekomunikační a informační technika

Student:	Bc. Tomáš Marek	***ID:***	83500
Ročník:	2	***Akademický rok:***	2008/2009

NÁZEV TÉMATU:

Softwarová podpora výuky kryptografických protokolů

POKYNY PRO VYPRACOVÁNÍ:

Prostudujte a popište v praxi nejčastěji používané kryptografické protokoly. Na tomto základě navrhněte a prakticky zrealizujte software pro podporu výuky uvedené problematiky. Software musí být spustitelné na běžném webovém prohlížeči. Kromě popisné části a ilustrativních příkladů musí obsahovat možnost interaktivních zobrazení a animací. Minimálním obsahem práce jsou základy protokolu TLS, protokolu 802.11i a kryptografický protokol v síti GSM.

DOPORUČENÁ LITERATURA:

[1] Stallings, W.: Cryptography and Network Security. Prentice Hall, Upper Saddle River 2006.
[2] Schneier, B.: Applied Cryptography. John Wiley, New York 1996.

Termín zadání:	9.2.2009	***Termín odevzdání:***	26.5.2009

Vedoucí práce: doc. Ing. Karel Burda, CSc.

prof. Ing. Kamil Vrba, CSc.
Předseda oborové rady

ABSTRAKT

Dokument obsahuje informace o autentizaci, šifrování, integritě dat, autentičnosti dat. Dále je zde popis známých kryptografických protokolů a jejich funkce, popřípadě jejich slabiny. Všechny tyto informace vedly k návrhu a realizaci softwaru pro podporu výuky kryptografických protokolů spustitelného na běžném webovém prohlížeči. Proto byla aplikace navržena jako webové stránky v PHP za využití i JavaSriptu a AJAXu. Tím je zajištěna i multiplatformnost a nezávislost na architektuře OS. Krom popisné a ilustrativní části aplikace obsahuje i interaktivní části a animace. V závěru textové části se nachází popis obsahu a funkcí výukového softwaru. Zdrojové kódy je možné nalézt na přiloženém CD.

KLÍČOVÁ SLOVA

TLS, 802.11i, 802.1X, IPsec, KERBEROS, GSM, kryptografické protokoly, autentizace, šifrování, zabezpečená komunikace.

ABSTRACT

Document contains informations about authentication, encryption, data integrity and data authenticity. Next part includes description of well know cryptography protocols, their functions and also their weaknesses. All of these acquired informations were used in concept and final software support for teaching of cryptography protocols, which is able to run on clasic web-browser. Thats why the application was designed as web PHP pages using JavaScript and AJAX, which ensures plaform and OS architecture independency. Besides the descripted and ilustrated part of application there are also interactive parts and animations. The last period contains description of education software and its functions. Source code can be found on the appended CD.

KEYWORDS

TLS, 802.11i, 802.1X, IPsec, KERBEROS, GSM, cryptography protocols , authentication, cryptography/encryption, secured comunikation.

MAREK, T. *Softwarová podpora výuky kryptografických protokolů*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2009. 69 s. Vedoucí práce doc. Ing. Karel Burda, CSc.

PROHLÁŠENÍ

Prohlašuji, že svou diplomovou práci na téma „Softwarová podpora výuky kryptografických protokolů“ jsem vypracoval samostatně pod vedením vedoucího diplomové 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 diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

V Brně dne ....

(podpis autora)

PODĚKOVÁNÍ

Děkuji vedoucímu diplomové práce doc. Ing. Karlu Burdovi, CSc. za velmi užitečnou metodickou pomoc, odborné vedení a cenné rady při zpracování diplomové práce. Dále bych chtěl poděkovat své matce za její podporu během celého studia především pak při zpracování této diplomové práce.

V Brně dne ....

(podpis autora)

OBSAH

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

ÚVOD

Šifrování provází lidstvo už od nepaměti a je nedílnou součástí moderní komunikace a programování. V dnešní době, kdy takřka veškerá komunikace probíhá elektronicky, je nezbytné využívat všechny možnosti šifrování a chránit tak naše data.

Náplní této diplomové práce je seznámit se s problematikou nejčastěji využívaných kryptografických protokolů, dále v nich prostudovat jejich hlavní rysy, např. autentizace stran, zajištění autentičnosti a integrity dat, zprostředkování bezpečného kanálu, předávání klíčů a odvozování tajných parametrů... Všechny tyto znalosti povedou k návrhu a poté realizaci softwaru na podporu výuky kryptografických protokolů. Vzhledem k obsahu, rozsahu, struktuře a maximální dostupnosti výukové části bude software primárně vytvořen jako internetové stránky za použití PHP.

Tato práce si klade za cíl vytvořit software vhodný pro použití při výuce zabývající se kryptografickými protokoly, zabezpečením komunikace nebo bezpečností vůbec. Software bude přehledný, strukturovaný, takže jeho obsah bude možné snadno později upravit či doplnit vzhledem k rychlému vývoji dané problematiky.

1 802.1X

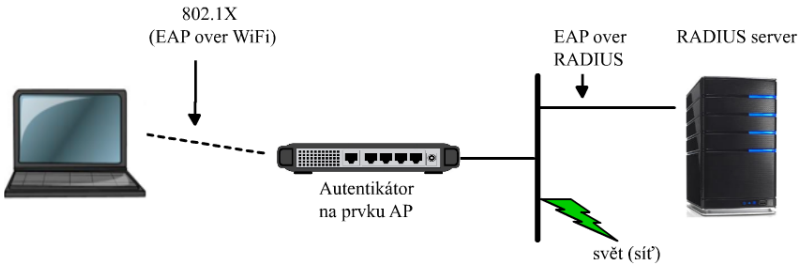
IEEE 802.1X je standard postavený na standardu EAP (Extensible Authentication Protocol). V dnešní době se požadavky na bezpečný přístup zvýšily, a proto bylo potřeba zajistit více než jednoduché jméno a heslo, které se využívalo v PPP (Point-to-Point Protocol). Proto byl vytvořen nový protokol pro ověřování a vyvedení této části mimo PPP, přičemž bylo zajištěno provázání obou protokolů.

EAP je alternativou k proprietárním ověřovacím systémům a umožňuje snadnou práci s hesly, tokeny i PKI (Public Key Infrastructure) certifikáty. Nezajišťuje ověřování jako takové, ale otevřený transportní mechanismus pro ověřovací systémy.

1.1 Funkce 802.1X

IEEE 802.1x používá tři komponenty, které mají své pojmenování:

- supplicant - uživatel nebo klient, který chce být ověřen
- authentication server - ověřovací server, typicky RADIUS server
- authenticator - zařízení mezi klientem a ověřovacím serverem - buď Access point nebo přepínač.

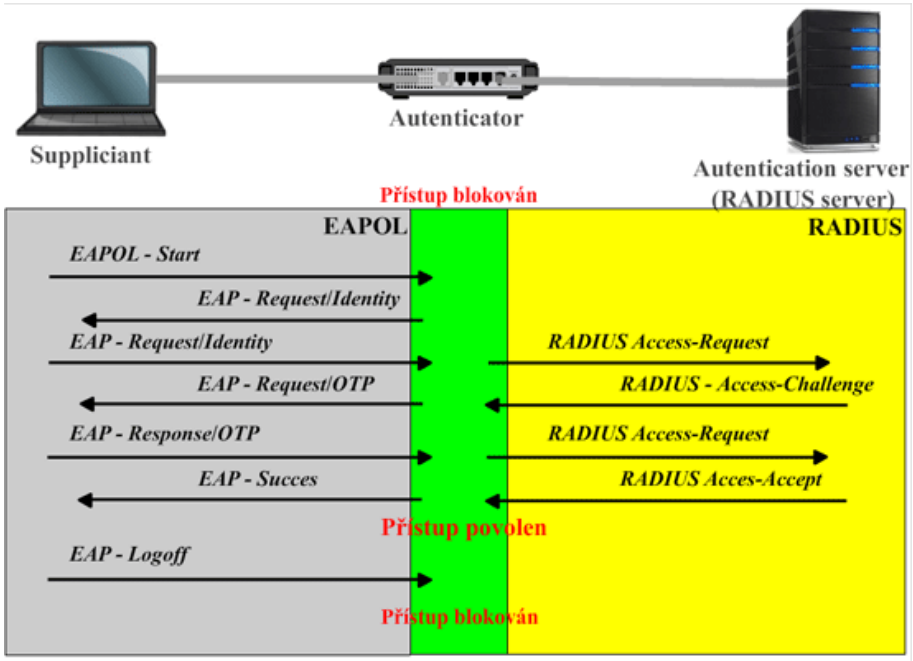


Obr. 1.1: Princip funkčnosti sítě s 802.1X

Jednou z klíčových vlastností 802.1X je to, že authenticator může být jednoduchý. Veškerá inteligence je v klientovi a ověřovacím serveru. To je ideální pro access pointy, protože jsou typicky poměrně jednoduché s malou pamětí a výkonem procesoru.

802.1X poskytuje různé možnosti, jako autentizace stanice, popřípadě serveru, pomocí certifikátů (PKI), předávání parametrů na odvození klíčů a předání hotových klíčů AP…

Protokol 802.1X je také nazýván zkratkou EAPOL (EAP over LANs). EAPOL není zvláště složitý a v základní implementaci není úplně bezpečný - zejména u bezdrátových sítí. Na následujícím obrázku je popis práce protokolu.Pro jednoduchost si představíme princip nejjednodušší metody s jednocestným ověřením.



Obr. 1.2: Přehled komunikace při spojení

1.2 Postup spojení

1. Supplicant může nezávisle na tom, zda dostal od authenticator požadavek, poslat informaci o tom, že je zde a chce se přihlásit do sítě („$EAPOL-Start$“).
2. Authenticator posílá na supplicant požadavek na ověření totožnosti kleinta („$EAP-Request/Identity$“) v okamžiku, kdy dostane požadavek („$EAPOL-Start$“) nebo detekuje aktivní link
   - v případě bezdrátové technologie se na access point snaží připojit nový klient (supplicant)
   - v případě přepínače se stav portu (s aktivovaným 802.1x ověřováním) změnil z down na up.

3. Supplicant posílá na authenticator informaci o své totožnosti („EAP-Response/Identity“), ten ji přebalí z EAP do RADIUS protokolu a posune ji na authentication server (RADIUS server).

4. Authentication server posílá zpět na authenticator výzvu pro ověření (*challenge*) - to je v podstatě řetězec znaků. Authenticator přebalí paket z formátu RADIUS do EAPOL a posílá jej na supplicant . Rozdílné ověřovací metody mají různé množství paketů v této části procesu (tzn. nemusí být pouze jednoduchý proces zobrazený na obrázku). EAP podporuje jak jednoduché ověření klienta, tak i silné vzájemné ověřování. Pro použití v bezdrátových sítích je vhodnější využívat vzájemné ověřování.

5. Supplicant na výzvu odpovídá authenticatoru (např. tak, že k výzvě přidá své heslo, provede hash a ten použije jako odpověď). Ten posunuje odpověď na authentication server.

6. Jestliže se supplicant prokáže řádnými údaji, authentication server odpoví zprávou o úspěšnosti, která je přeposlána na supplicant. Authenticator umožní přístup do sítě - s možností restrikcí na základě atributů od authentication server. Například přiřadí supplicant do konkrétní VLAN nebo nastaví filtrovací pravidla.

Důležité je si povšimnout, že authenticator funguje jen jako prostředník, který vždy přijatou informaci přečte z přijatého paketu a uloží ji do paketu nového dané komunikace. Tzn. když přijme paket TCP od supplicanta, „vytáhne“ přijatou informaci a uloží ji do paketu typu RADIUS.

EAPOL má další možnosti, např.:

- Zpráva ukončení práce od supplicant („$LOGFF$“), pokud se supplicant odhlásí nekorektně (např. link na přepínači jde do stavu down nebo access point ztratí spojení s klientem), je na aktivním prvku ukončeno oprávnění a supplicant se musí opět ověřit.

- 802.1X také umožňuje definovat časovou prodlevu pro opětovné ověřování, supplicant se tak musí periodicky prokazovat.

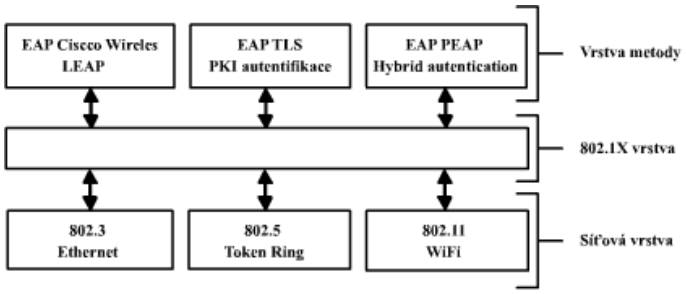
1.3 Typy ověřování

Již bylo řečeno, že EAP protokol zajišťuje pouze transportní mechanismus pro ověřování. Umožňuje to poměrně snadno vytvářet nové modifikace protokolu - ten se

v principu nemění, pouze ověřovacímu mechanismu musí rozumět supplicant a authentication server.

Architekturu si lze představit podle následujícího obrázku 1.3Má tyto vrstvy:

- Linková vrstva - uzpůsobení konkrétní LAN technologii (definice specifických rámců)
- 801.1X vrstva - zajištění pravidel pro komunikaci komponenty systému (supplicant, authenticator a authentication server)
- vrstva ověřovací metody - řeší konkrétní metodu ověření uživatele, příslušných metod je mnoho, na obrázku je jich jmenována část, další jsou v popisu metod.



Obr. 1.3: Vrstvový model architektůry 802.1X

Uvádím ty nejznámější a nejzajímavější.
EAP-MD5 - (RFC 1994, RFC 2284) pro ověřování je používáno uživatelské jméno a heslo, ty jsou pro zajištění pravosti hašovány pomocí MD5. Jde o původní specifikaci s jednocestným ověřováním. Tento způsob se nejvíce blíží tomu, co bylo představeno na schématu. Další alternativy používají složitější mechanismus výměny informací a v navazovací sekvenci je více kroků.

EAP-OTP - (RFC 2289) (One-Time Password) využívá pro ověřování některý ze systémů typu CRYPTOCard Token, RSA SecureID, SecureComputing SafeWord Token ...

EAP-GTC - (Generic Token Card) - obdoba EAP-OTP.

EAP-TLS - (RFC 2716) (Transport Level Security) - pro ověřování použito PKI a TLS mechanismus ověřování je založen na použití certifikátů, umožňuje vzájemné ověření klienta a sítě (tedy i síť prověřuje klientovi svou pravost). Klíče jsou generovány dynamicky. Tato verze byla zvolena jako základní pro implementaci ve

Windows (2000 a XP).

EAP-LEAP - (Lightweight Extensible Authentication Protocol) proprietární ověřovací mechanismus využívané v Cisco podporuje vzájemné ověření klienta a sítě. Je obdobou EAP-TLS, ale namísto certifikátů používá jméno a heslo. Zajišťuje mechanismus dynamického generování klíčů pro WEP.

EAP-TTLS - (Tunneled TLS) rozšíření modifikace EAP-TLS, používá TLS pro navázání spojení s ověřovacím serverem a vytvoření tunelu pro druhý („vnitřní“) ověřovací algoritmus.

EAP-PEAP - (Protected EAP) podobný TTLS, zajišťuje TLS k vytvoření tunelu pro druhý ověřovací algoritmus, ten je typu EAP.

2 IPSEC

IPsec (IP security) je rozšíření IP (Internet Protocol) protokolu, které poskytuje bezpečnost pro IP protokol a protokoly vyšších vrstev. Nejdříve se vyvinul pro nový standard IPv6 a následně byl zpětně implementován na IPv4. IPsec architektura je popsaná v RFC2401.

IPsec užívá dva rozdílné protokoly - AH (Authentication Header) a ESP (Encapsulated Security Payload) - aby zajistil ověřování identity, integritu a důvěryhodnost komunikace. Může zajišťovat buď autentičnost celého IP paketu(AH protokol), nebo zajišťuje důvěrnost a autentičnost pouze přenášených dat (ESP protokol). Provozní varianty příslušných protokolů se nazývají tunelový mód a transportní mód. V transportním módu je IP paket plně zapouzdřený do nového IP datagramu, který používá IPsec protokol, tzn. k původnímu paketu je připojeno nové záhlaví. V tunelovém módu je k původnímu paketu připojeno nové záhlaví. Transportní mód se používá pro přenos mezi koncovými zařízeními, zatímco tunelový mód se využívá pro přenos paketů mezi branami různých sítí.

Pro ochranu neporušenosti (integrity) IP paketů IPsec protokoly používají HMAC (Hash Message Authentication Code). Aby IPsec protokoly získaly tento HMAC, užívají hash algoritmy jako MD5 a SHA-1 a vypočtou hash na základě tajného klíče (secret key) a obsahu IP diagramu, výstupem je potom autentizační kód IVC (Integrity Check Value). Tento HMAC je potom zahrnut do hlavičky IPsec protokolu a příjemce paketu může kontrolovat HMAC, pokud má přístup k tajnému klíči.

Výpocet ICV pomocí HMAC:

1. Klíč k se doplní nulami na délku 512 bitů. Tento řetězec se označí K.
2. $h1 = H[(K \oplus ipad) \| D]$, kde $ipad$ je řetězec z 64 bajtů $0x36$.
3. $h2 = H[(K \oplus opad) \| h1]$, kde $opad$ je řetězec z 64 bajtů $0x5C$.
4. ICV je prvních 96 bitů $h2$.

Pro ochranu důvěryhodnosti IP datagramů IPsec protokoly užívají standardní symetrické šifrovací (encryption) algoritmy. IPsec standard požaduje implementaci NULL a DES (Data Encryption Standard). Dnes jsou však obvykle užívané silnější algoritmy jako 3DES, AES (Advanced Encryption Standard) a Blowfish. Šifrovaná data se musí doplnit výplňovými bajty (PAD) tak, aby celková délka dat i s výplní

byla násobkem délky bloku použité šifry.

Pro ochranu proti útokům typu DoS (Denial of Service) IPsec protokoly užívají tzv. okno (sliding window). Každému paketu je přiřazeno sekvenční číslo (Sequence Number) a je přijatý pouze jestliže číslo paketu je v rámci daného okna nebo novější. Starší pakety jsou okamžitě zrušeny. Toto chrání proti útokům založeným na opakování paketů (replay attacks), kdy útočník zaznamená původní pakety a přehraje je později.

2.1 Protokol IKE

Kritické problémy pro systémové administrátory speciálně představuje výměna klíčů: Jak vyměnit tajné symetrické klíče, když ještě není žádné šifrování?

Klíče mohou doručit ještě kurýři, ale to je neefektivní. Aby se vyřešil tento problém, byl vyvinut IKE (Internet Key Exchange) protokol založený na asymetrické kryptografii.Protokol IKE funguje prakticky stejně jako Handshake protocol v TLS (Transparent Layer Security). Tento protokol v první fázi ověřuje identitu protějšků komunikace. Ve druhé fázi jsou vyjednané SAs a jsou vybrané tajné symetrické klíče např. pomocí výměny klíčů metodou Diffie Hellmana. IKE protokol se potom dokonce stará o periodickou změnu tajných klíčů, aby se zajistila důvěryhodnost.

Aby protější strana komunikace mohla zapouzdřit a rozpouzdřit IPsec pakety, potřebuje nějaký způsob, jak uložit tajné klíče, algoritmy a IP adresy zahrnuté v komunikaci. Všechny tyto parametry potřebné pro ochranu IP paketů jsou uložené v SA (Security Association). SAs (tj. mn. č. pro SA) jsou uloženy v databázi SAD (Security Association Database).

Každé SA definuje následující parametry:

- Zdrojovou a cílovou IP adresu výsledné IPsec hlavičky. Toto jsou IP adresy protějších stran IPsec komunikace chránících pakety.
- IPsec protokol (AH nebo ESP), někdy je také podporovaná komprese (IPCOMP).
- Algoritmus a tajný klíč užívaný IPsec protokolem.
- SPI (Security Parametr Index). Tj. 32 bitové číslo, které identifikuje SA.

Některé implementace SAD databáze umožňují, aby byly ukládány i další parametry:

- IPsec mód (tunelový nebo transportní)
- Velikost okna (sliding window) pro ochranu proti útokům založeným na přehrání zachycených dat.
- Doba SA existence.

Poněvadž SA definuje zdrojové a cílové IP adresy, může chránit jen jeden směr datového provozu v plně duplexní IPsec komunikaci. Aby se chránily oba směry, IPsec potřebuje dva jednosměrné SAs.

SA pouze specifikuje jak IPsec chrání datový provoz. Další informace je potřebná pro definování toho, který datový provoz chránit. Tato informace je uložena v SP (Security Policy), což je uloženo v databázi SPD (Security Policy Database).

SP obvykle určuje následující parametry:

- Zdrojová a cílová adresa paketů, které se mají chránit. V transportním módu jsou to stejné adresy jako v SA. V tunelovém módu se mohou lišit.
- Protokol a port, jež se mají chránit. Některé IPsec implementace nedovolují definice specifických protokolů, které se mají chránit. V tomto případě je chráněn veškerý datový provoz mezi zmíněnými adresami.
- SA, které se má použít pro ochranu paketů.

2.2 IPsec protokoly

Rodina IPsec protokolů se skládá ze dvou protokolů: AH (Authentication Header) a ESP (Encapsulated Security Payload). Oba jsou nezávislé protokoly. AH je IP protokol číslo 51 a ESP je protokol číslo 50.

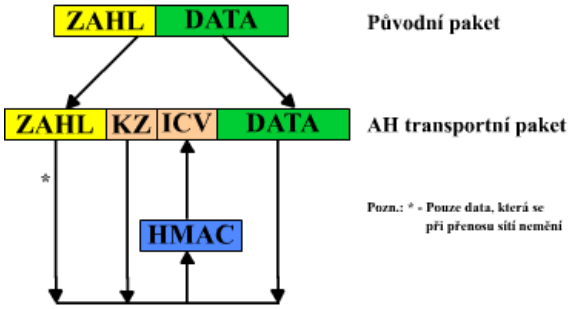
2.2.1 AH - Authentication Header

AH protokol chrání integritu a autentičnost IP paketu. Aby se toho dosáhlo, AH protokol počítá HMAC. Když se počítá HMAC, AH protokol vychází z tajného klíče, užitečné části paketu a neměnných částí IP hlavičky jako jsou IP adresy. Potom k paketu přidává AH hlavičku (viz obrázek 2.1).

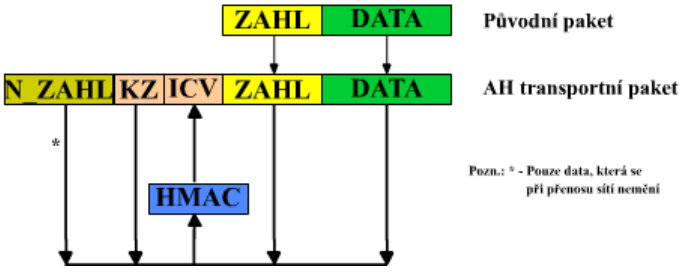
Next Header 1B	Payload length 1B	Reserved 2B
Security Parameter Index (SPI) 4B		
Sequence Number (Replay Defense) 4B		
Hash Message Authentication Code (HMAC) 12B		

Obr. 2.1: Hlavička AH protokolu

AH hlavička je 24 bajtů dlouhá. První bajt je políčko *Next Header*. Toto políčko určuje protokol následující hlavičky. V tunelovém módu je zapouzdřen celý IP datagram, proto hodnota tohoto políčka je 4. Když se zapouzdřuje TCP datagram v transportním módu, odpovídající hodnota je 6. Další bajt specifikuje délku užitečné části paketu. Toto políčko je následované dvěma rezervovanými bajty. Další dvojité slovo udává 32 bitů dlouhé SPI. SPI specifikuje SA, které se má použít pro rozpouzdření paketu. 32 bitů dlouhé sekvenční číslo (*Sequence Number*) chrání proti útokům založeným na opětovném přeposlání zachycených dat. Konečně posledních 96 bitů obsahuje HMAC. HMAC chrání integritu paketů, poněvadž pouze protější strany komunikace znající tajný klíč mohou vytvořit a kontrolovat HMAC.



Obr. 2.2: Protokol AH, transportní varianta

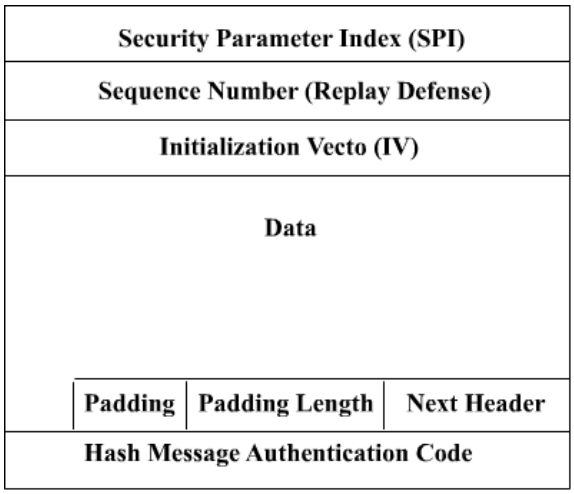


Obr. 2.3: Protokol AH, varianta tunelová

AH nedovoluje NAT (Network Address Translation), poněvadž AH protokol chrání IP datagram včetně neměnných častí IP hlavičky jako jsou IP adresy. NAT nahrazuje IP adresu v IP hlavičce jinou (obvykle zdrojovou) IP adresou. Po této náhradě již není HMAC platný. Toto omezení obchází implementace rozšíření IPsec protokolu NAT-Traversal.

2.2.2 ESP - Encapsulated Security Payload

ESP protokol může zajistit jak integritu paketu pomocí HMAC, tak i důvěryhodnost pomocí šifrování. Po zašifrování paketu a výpočtu HMAC se vytvoří ESP hlavička a přidá se do paketu. ESP hlavička se skládá ze dvou částí a je prezentována na obrázku.

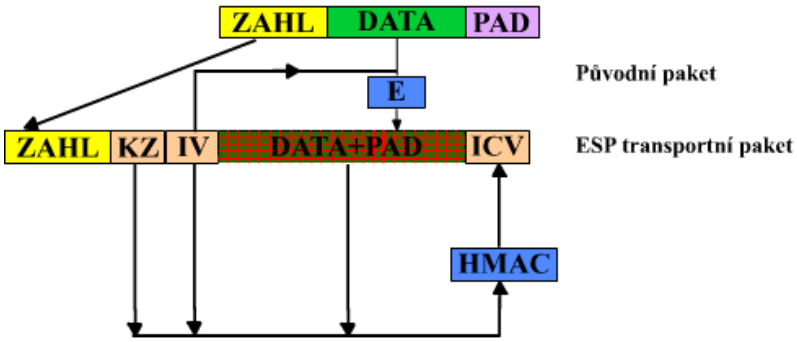


Obr. 2.4: Hlavička ESP protokolu

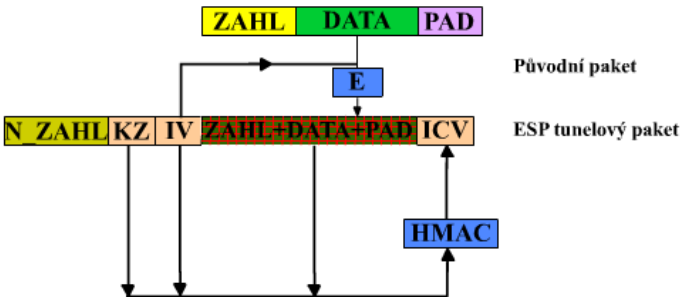
První dvojité slovo v ESP hlavičce specifikuje SPI. SPI udává SA, jež se má použít pro rozpouzdření ESP paketu. Druhé dvojité slovo obsahuje sekvenční číslo (*Sequence Number*). Toto sekvenční číslo chrání proti útokům založeným na opětném přehrání zachycených dat. Třetí dvojité slovo určuje IV(Initialization Vector), který je používán v šifrovacím procesu. Symetrické šifrovací algoritmy jsou náchylné k frekvenčnímu útoku (frequency attack), pokud není užíván žádný IV. IV zajišťuje, že dvě stejné užitečné části paketů vedou ke dvěma různým zašifrovaným užitečným částem paketů.

IPsec pro šifrovací proces používá blokové šifry. Jestliže délka užitečné části není násobkem délky bloku, může být potřeba užitečnou část paketu vyplnit „bajtovou vycpávkou“ (PAD). Potom je přidaná délka vycpávky. Za délkou vycpávky následuje *Next Header* o délce 2 bajty, který specifikuje následující hlavičku. Nakonec je do ESP hlavičky přidán 96 bitů dlouhý HMAC zajišťující integritu paketu. HMAC bere

do úvahy pouze užitečnou část paketu. IP hlavička není zahrnuta do výpočetního procesu.



Obr. 2.5: Protokol ESP, transportní varianta



Obr. 2.6: Protokol ESP, varianta tunelová

Použití překladu adres NAT proto neporuší ESP protokol. Ve většině případů však stále ještě není NAT možný ve spojení s IPsec technologií. V tomto případě NAT-Traversal nabízí řešení zapouzdřením ESP paketů do UDP paketů.

3 KERBEROS

Kerberos poskytuje sofistikované mechanismy pro autentizaci a zabezpečení síťové komunikace. Systém Kerberos je definován standardem IETF RFC 1510 a tvoří základní autentizační prvek v řadě komerčních i open-source systémů, především autentizaci ve Windows. Kerberos je navržen tak, aby zajišťoval silné zabezpečení současně s jednoduchým uživatelským rozhraním. Způsob autentizace je v systému Kerberos založen na použití tzv. „lístků“ vydávaných KDC(centrální autentizační server), který spravuje databázi všech uživatelů. Lístky jsou analogické např. certifikátům veřejných klíčů, které jsou známé z prostředí PKI, ale na rozdíl od nich jsou kerberovské lístky a všechny kerberovské protokoly založeny výhradně na použití symetrické kryptografie, tj. hesla sdíleného mezi uživatelem a centrálním autentizačním serverem. Kerberovské lístky mají také kratší dobu platnosti (zpravidla deset hodin) a vždy obsahují informaci, pro kterou konkrétní koncovou službu jsou určeny.

Vedle podpory autentizace poskytuje mechanismus kerberovských lístků také podporu tzv. principu *single sign-on*, který umožňuje uživatelům pohodlné použití prostředků aniž by byli zbytečně zatíženi složitými bezpečnostními procedurami. Uživatel se totiž autentizuje vůči serveru Kerbera pouze jednou a získá tak základní lístek (tzv. Ticket Granting Ticket (TGT)), který se dále použije k získání dalších lístků pro přístup k službám v systému, které vyžadují autentizaci. Při vzdáleném přihlašování na jiné stroje přenese také TGT lístek a uživatel tak má stále možnost, jak se autentizovat. Veškeré operace kromě prvotní autentizace však již probíhají transparentně bez zásahu uživatele, který tak není zdržován ve své činnosti.

Autentizace pracuje na bázi důvěrnosti třetí straně, která se jmenuje KDC (key distribution center), která se skládá ze dvou logických oddělených prvků : Authentication Server (AS) zajišťující autentizaci a Ticket Granting Server (TGS), který zajišťuje distribuci lístků.

KDC také obsahuje databázi tajných klíčů, každé entity nebo sítě, klienta nebo serveru, které jsou známé jen jemu a KDC.

3.1 Postup autentizace

Předpokládejme prostředí klient-server, ve kterém klient (program, služba nebo proces), který je aktivní v relaci přihlášení konkrétního uživatele, požaduje přístup k určité službě. Může se jednat např. o otevření složky nebo souboru, nebo zpracování určitého databázového dotazu. V bezpečnostním modelu protokolu Kerberos každé

takové akci předchází autentizace.

Jsou použity tyto prvky:

- AS = Authentication Server
- SS = Service Server
- TGS = Ticket-Granting Server
- TGT = Ticket-Granting Ticket

3.1.1 Základní přihlášení uživatele

1. Klient pošle nezašifrovanou zprávu do AS, což je vlastně žádost o lístek pro přidělování žádostí TGT.
   "Uživatel XX by chtěl využít služby YY." Žádný tajný klíč nebo heslo nejsou zaslány do AS. Nicméně AS je schopno si vygenerovat tajný klíč uživatele zhašováním uložených parametrů uživatele z vlastní databáze na KDC.
2. Zadané parametry se zpracují jednocestnou funkcí (nejčastěji hash) a vytvoří se tak tajný klíč klienta (secret key klient). Tento klíč je tzv. dlouhodobý.

3.1.2 Autentikace uživatele

1. Uživatel zadá uživatelské jméno a heslo klientskému počítači.
2. AS zkontroluje, zda-li je klient v databázi. Pokud ano, AS pošle zpět tyto dvě zprávy:
   - Zpráva A: Client/TGS Session Key zašifrovaný tajným klíčem uživatele
   - Zpráva B: Ticket-Granting Ticket(obsahuje uživatelovo ID,uživatelovu síťovou adresu, časové omezení lístku a client/TGS session key) zašifrovaný secret key TGS.
3. Jakmile klient obdrží zprávy A a B, dešifruje zprávu A a získá tak *Client/TGS Session Key*. Tento klíč je použit pro budoucí komunikaci s TGS.// Uživatel nemůže dešifrovat zprávu B, protože je zašifrována *TGS's secret key*. V tomto okamžiku má klient dostatek informací, aby se autentizoval u TGS.

Poznámka: *Klíč relace - TGS Session Key je určen pouze pro komunikaci mezi klientem a KDC a je platný po celou dobu relace autentizace. Po získání tohoto klíče není dále nutné, aby klient komunikoval s KDC za pomocí svého hlavního tajného*

klíče. Tímto mechanizmem je minimalizováno použití nejcitlivější informace, kterou je dlouhodobý hlavní klíč klienta.

3.1.3 Uživatel se autentizuje u TGS - Client Service Authorization

1. Při žádosti uživatel o službu pošle tyto dvě zprávy do TGS:
   - Zpráva C: Obsahuje zprávu B od TGT a ID žádané služby.
   - Zpráva D: Autentikátor (obsahuje ID klienta a časové razítko), zašifrované *Client/TGS Session Key*
2. Poté, co klient obdrží zprávy C a D, TGS vyjme zprávu B ze zprávy C. Dešifruje ji za pomocí TGS secret key. Tím získá client/TGS session key. Použitím tohoto klíče TGS dešifruje zprávu D(autentikátor) a pošle následující zprávy:
   - Zpráva E: Client-to-server ticket (obsahuje uživatelovo ID, uživatelovu síťovou adresu, dobu platnosti a *Client/Server Session Key*) zašifrovaný service's secret key.
   - Zpráva F: Client/server session key zašifrovaný Client/TGS Session Key.

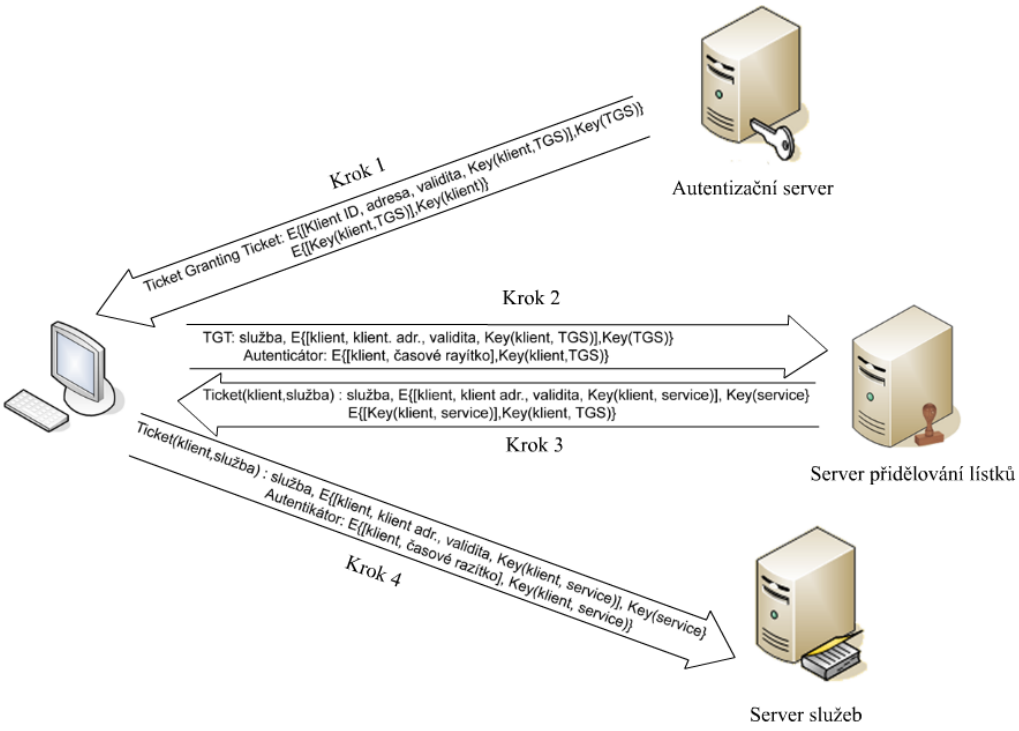
3.1.4 Uživatel žádá o službu - Client Service Request

1. Po obdržení zpráv E a F od TGS má klient dostatek informací, aby se autentizoval u SS. Klient se spojí s SS a pošle mu tyto dvě zprávy:
   - Zpráva E z předešlého kroku (client-to-server lístek zašifrovaný service's secret key)
   - Zpráva G: nový Authenticator, který obsahuje uživatelovo ID, časové razítko a je zašifrovaný client/server session key.
2. SS dešifruje lístek za použití vlastního secret key, aby získal z přijaté zprávy Client/Server Session Key. Po dešifrovaní zprávy autentikátor pošle klientovi následující zprávu :
   - Zpráva H: časové razítko získané ze zprávy autentikátor a přičte 1, zašifruje Client/Server Session Key.
3. Klient dešifruje potvrzení klíčem Client/Server Session Key a zkontroluje, zda časové razítko bylo korektně upraveno. Pokud ano, klient může důvěřovat serveru a může bezpečně zažádat server o chtěnou službu.

4. Server poskytne požadovanou službu klientovi.

Poznámka: *Autentizace nemusí být jednosměrná. Také klient muže požadovat, aby server prokázal svou identitu. Takový případ se označuje jako vzájemná autentizace.*

Z popisu je patrná další výhoda protokolu Kerberos. Služba KDC ani žádná jiná služba nemusí uchovávat kopie klíčů relací. Tyto kopie služby získávají od klientů formou lístku - KDC prostřednictvím TGT a ostatní služby formou lístku relací (Session Tiket).



Obr. 3.1: Přehled komunikace

3.2 Nevýhody

- Jednoduchý bod selhání: Je vyžadována dostupnost KDC po celou dobu spojení. Pokud by Kerberos server byl nefunkční, nikdo by se nemohl přihlásit. Toto se může odstranit použitím více Kerberos serverů a jejich zpětnou vazbou.
- Kerberos požaduje, aby byli klienti časově synchronizovaní se serverem. Lístky mají časové omezení a pokud hodiny uživatele nebudou synchronizované s kerberos servrem, autentizace bude neúspěšná. Defaultní nastavení je vyžadováno,

aby se hodiny nelišily více jak o 10 minut. V praxi se pak používá Network Time Protocol, který udržuje klientské hodiny synchronizované.

- Pokud budou všechny klíče uživatelů uloženy na centrálním serveru a ten bude kompromitován, budou tak kompromitovány všechny uživatelské tajné klíče.
- Kompromitující uživatel může kompromitovat uživatelské heslo.

4 802.11I

IEEE 802.11i je dodatek standardu 802.11 specifikující bezpečnostní mechanismy pro bezdrátové sítě (WiFi). Návrh normy byl potvrzen v červnu 2004 a nahradil předchozí bezpečnostní specifikaci – WEP (Wired Equivalent Privacy), u které se projevila nízká zabezpečovací schopnost.

Konečná verze normy 802.11i sestává z několika částí, z nichž nejdůležitější jsou dva nové zabezpečovací protokoly – TKIP a CCMP. Dále také využívá systém kontroly přístupu k síti, definovaného podle normy 802.1X (v rámci návrhu normy 802.11i se tedy počítá s využitím normy 802.1X (viz kapitola 1).

Jelikož u standardu 802.11 jsou rozdílně spravovány unicastové a broadcastové přenosy, je pro každý typ přenosu definován proces vyjednávání vhodného zabezpečovacího protokolu a systému výměny klíčů.

4.1 Historie

V roce 1999 byly WiFi sítě zabezpečovány pomocí WEP.

IEEE 802.11i program aktuálně začal dohromady s Duality of Service and Security pojmenovaného IEEE 802.11e. Nicméně rychle se stalo zřejmé, že kvůli markantnímu rozšíření WiFi bezpečnost potřebuje vlastní specifikace, a tak se stávající norma rozdělila na IEEE 802.11e, která pokračovala v práci na Quality of Service (QoS), a IEEE 802.11i, která se soustředila na bezpečnost.

Ještě před schválením normy 802.11i byl Wi-Fi Alliancí dodatečně uveden standard WPA (Wi-Fi Protected Access) v roce 2003, který zavedl dočasné řešení vůči bezpečnostním problémům u WEP. V rámci WPA byla implementována podmnožina později schváleného standardu 802.11i. Plná implementace WPA podle 802.11i je nazývána WPA2.

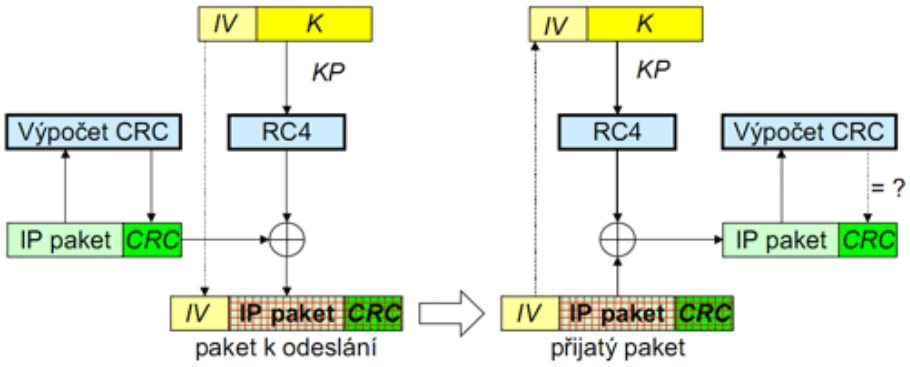
4.2 WEP

WEP používá proudovou šifrovací metodu RC4 pro utajení informací a pro ověření jejich autentikace používá metodu CRC-32 kontrolního součtu.

Takzvaný 64-bitový WEP používá 40bitový klíč, ke kterému je připojen 24-bitový inicializační vektor a dohromady tak tvoří 64-bitový RC4 klíč. 128-bitový WEP používá 104-bitový klíč, ke kterému je připojen 24-bitový inicializační vektor a dohromady tak tvoří 128-bitový RC4 klíč. Někteří výrobci bezdrátových zařízení

poskytují i 256-bitový WEP.

Bohužel je ale známo mnoho slabostí WEP zabezpečení. Prvním nedostatkem se ukázala už samotná délka klíče a fakt, že tento klíč se používal pro celou WiFi síť s ručním nastavením. WEP mohl být dokonce prolomen pasivním útokem za použití veřejně dostupných prostředků. Další velkou nevýhodou se ukázalo použití CRC-32 kontrolního součtu na ověřování dat. Tento kontrolní součet lze poměrně snadno spočítat i bez znalosti klíče a pozměňovat tak původní zprávy. Společně s kolizemi inicializačních vektorů a možností útoků pomocí zasílání pozměněných paketů dělá z WEP velmi slabé zabezpečení.



Obr. 4.1: Schéma funkčnosti WEP

4.2.1 WEPplus

Někdy označováno jako WEP+. Vylepšení původního WEP zabezpečení od Agere Systems, které se snaží odstranit takzvané slabé inicializační vektory. Jsou to ty inicializační vektory, pomocí kterých může útočník velmi rychle spočítat použitý klíč, a tak se může do sítě zabezpečené pomocí WEP připojit. Pokud ovšem není WEPplus na všech komunikujících stranách v bezdrátové síti, nemá toto zabezpečení výhody oproti běžnému WEP.

4.2.2 WEP2

Vylepšení původního WEP zabezpečení, které se snaží odstranit bezpečností díry. WEP2 rozšířil inicializační vektory a zesílil 128-bitové šifrování. Byl použit typicky na zařízeních, na kterých nebylo možné provozovat novější WPA nebo WPA2 zabezpečení. Nicméně WEP2 má stejné bezpečnostní problémy jako WEP, jen útočníkovi zabere více času a to dokonce jen lineárně a ne exponenciálně, jak se předpokládalo.

4.3 WPA

Data jsou zašifrována pomocí proudové šifrovací metody RC4 se 128-bitovým klíčem a 48-bitovým inicializačním vektorem (IV). Zásadní vylepšení oproti WEP zabezpečení spočívá v použití TKIP (Temporal Key Integrity Protocol), což je protokol dynamicky měnící klíče. Společně s mnohem delšími inicializačními vektory tak odolává útokům, jimž podléhal WEP.

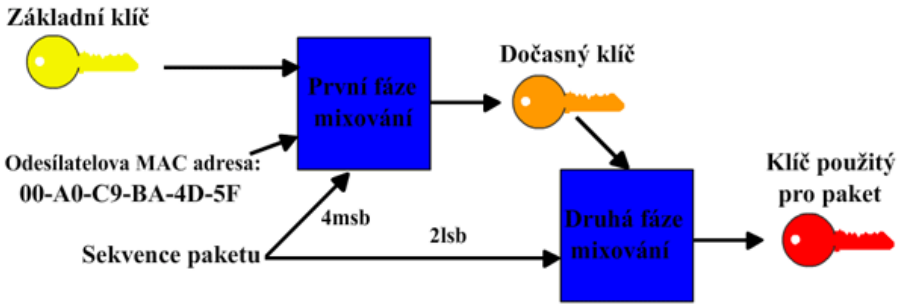
Kromě autentizace a šifrování WPA také vylepšuje kontrolu správnosti dat, tedy integritu. WEP používá metodu cyklického kontrolního součtu CRC-32, která je sama o sobě málo bezpečná, protože je možné pozměnit zprávu a kontrolní součet bez znalosti WEP klíče. WPA používá bezpečnější MAC (Message Authentication Code), zde nazvanou MIC (Message Integrity Code), konkrétně algoritmus nazvaný *Michael*. MIC metoda použitá v WPA zahrnuje počítadlo rámců, které chrání před útoky snažícími se zopakovat předchozí odposlouchanou komunikaci.

4.3.1 TKIP

TKIP je zabezpečovací protokol navržený pro zlepšení zabezpečení starších produktů, které implementují původní WEP protokol, soustředící se na nepřítomnost silného kontrolního kódu (WEP používá pouze CRC32, který je možno celkem jednoduše obejít). TKIP zavádí implementaci kontrolního součtu pod kódovým názvem Michael, který zařízením umožňuje ověřit fakt, že příchozí pakety skutečně pocházejí od toho, kdo se prezentuje jako odesílatel. Zároveň umožňuje kontrolu, zda obsah paketu nebyl po cestě změněn. Algoritmus Michael představuje to nejsilnější, co mohli autoři WPA použít při zachování kompatibility se staršími síťovými kartami. Díky nevyhnutelné slabosti algoritmu Michael obsahuje WPA speciální ochranný mechanismus, který detekuje pokus o prolomení TKIP a dočasně blokuje komunikaci s útočníkem.

U původního WEP protokolu se konfigurací stanovil pevný šifrovací klíč (popř. se získal pomocí EAP, definovaného v rámci 802.1X), nazývaný též PMT (Pairwise Master Key). Tento sdílený klíč se používal po celou dobu přenosu mezi AP a všemi klienty.

TKIP s využitím tzv. „mixovací“ funkce zajišťuje ochranu před útoky na získání klíče, viz Obr4.2. S pomocí této funkce vytváří jedinečný klíč pro každý přenosový rámec a odstraňuje tak slabé místo v algoritmu RC4, využívaného původním WEP protokolem.



Obr. 4.2: Princip mixovací funkce TKIP

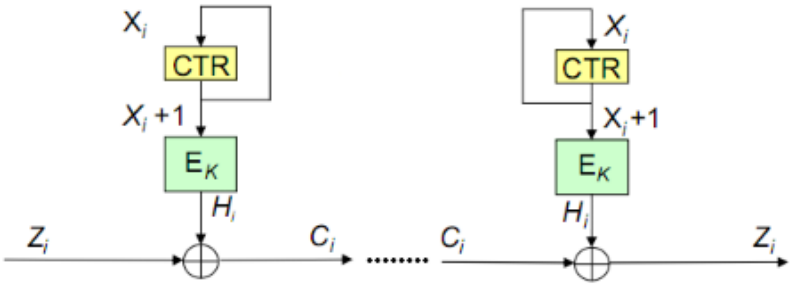
Díky prodloužení klíčů a inicializačních vektorů, snížení počtu zaslaných paketů s příbuznými klíči a systému ověřujícímu zprávy je těžké WPA prolomit.

4.4 WPA2

4.4.1 CCMP (Counter-Mode/CBC-MAC Protocol)

Jelikož TKIP byl vytvořen pouze jako rozšíření pro stávající zařízení (funguje jako nadstavba nad WEP), CCMP je definován jako silnější verze pro novější zařízení, která byla vytvořena i pro uspokojení potřeb Federal Information Processing Standards (FIPS).

V prvním zpracování se použilo AES v módu OCB (Offset Codebook), ale kvůli potížím s několika členy vývojového týmu se přešlo na AES – čítačový mód (AES – Counter-Mode).

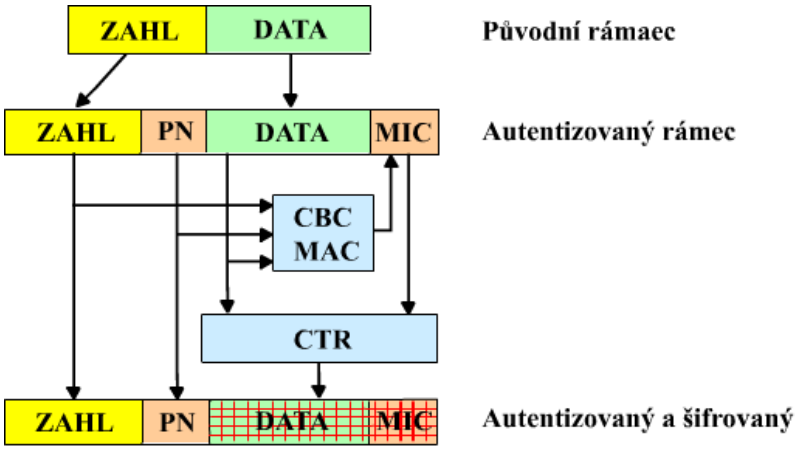


Obr. 4.3: Blokové schéma Counter-Mode

CCMP je protokol, který zajišťuje jak autentikaci paketů, tak jejich šifrování. Pro zajištění důvěrnosti používá šifrování AES. Pro zajištění autentikace a integrity dat používá CBC-MAC (Cipher Block Chaining Message Authentication Code). Podle 802.11i používá CCMP 128-bitový klíč a velikost šifrovaného bloku je 128 bitů.

Výsledný CCMP paket je potom o 16 oktetů delší než standardní nešifrovaný 802.11 paket (8 oktetů CBC-MAC, 6 oktetů náhodné číslo a dva oktety pro další režii). Delší paket se ale ukázal jako dobrá výměna za mnohem lepší bezpečnost.

CCMP chrání ta pole rámce, která nejsou standardně šifrována. Jedná se zejména o adresu zdroje a cíle paketu. Takto zabezpečené části se nazývají AAD (Additional Authentication Data).



Obr. 4.4: Blokové schéma funkčnosti WPA2 CCMP

4.4.2 Vyjednávání o použití vhodného protokolu

Protože v rámci 802.11i je definováno více zabezpečovacích protokolů, poskytuje norma klientovi a AP prostředky, pomocí kterých se mohou dohodnout na použití konkrétního protokolu. Volba se odvíjí např. podle vyžadovaného typu přenosu, nebo od dodatečně zjištěných bezpečnostních parametrů AP nebo klienta.

Informace o bezpečnostních parametrech v síti zasílá AP ve svých signálních (*beacon*) rámcích, popřípadě je sděluje zájemcům (*probe request – probe response*). Informace, které AP pro tyto účely poskytuje, jsou následující:

- group ciphersuite – sada protokolů, kterou lze použít při zasílání broadcast zpráv
- pairwise ciphersuite list – seznam protokolů, který lze použít při párové komunikaci klient – AP (unicast)
- authentication and key management suite – jedná se o informace dostupné v případě, že je znám sdílený klíč nebo je používán 802.1X

Pokud si klient zjistí parametry sítě, zvolí vhodnou kombinaci a svůj výběr zašle v požadavku k AP. Výběr se musí shodovat s některými z položek dostupných v nabídce AP. V opačném případě AP zašle zamítavou odpověď. Až do této chvíle není proces vyjednávání zabezpečený. Zabezpečeným se stává v okamžiku úspěšné výměny klíčů podle některé metody použité z EAPOL (viz kapitola 1).

4.4.3 Hierarchie klíčů

V procesu výměny klíčů se podle IEEE 802.11i využívá více druhů klíčů v závislosti na typu komunikace. Daný druh klíče rozdělujeme na sadu podklíčů využitelnou pro další potřeby. Jsou definovány dva základní druhy:

- Pairwise key – pro unicastové přenosy, viz Obr4.5
- Group key – pro multicast a broadcast přenosy, viz Obr4.6

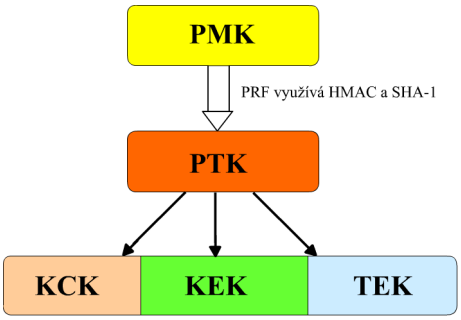
Původní norma 802.1X definuje speciální rámec určený pro výměnu klíčů. V normě 802.11i jsou ale definovány jiné metody výměny klíčů. Pro unicast je použit tzv. 4-way handshake (viz kapitola 1) a pro multicast/broadcast je použit tzv. group key handshake (viz kapitola 2).

1. **Pairwise key**
   Výchozím bodem této hierarchie je PMK (Pairwise Master Key). Pokud je použit 802.1X protokol (EAP), potom tento klíč dodá autentikační server. V případě použití sdíleného klíče se PMK přiřazuje na základě předaného hesla. Následně se s využitím pseudonáhodné funkce vytvoří z PMK a několika dalších parametrů (viz níže) nový klíč – PTK (Pairwise Transient Key). Výsledný PTK klíč se dále rozděluje na 3 klíče (viz Obr4.5):

   - Potvrzovací klíč – KCK (EAPOL-key Confirmation Key) – využívá se pro ověření původu dat
   - Šifrovací klíč – KEK (EAPOL-key encryption key) – využívá se pro šifrování dat
   - Dočasný klíč PTK (Temporal Key) – ten využívají zabezpečovací protokoly (kontrolní součty)

   Při tvorbě se berou v potaz tyto parametry :

   - MAC adresa žadatele (klient)
   - MAC adresa ověřovatele (Access Point)
   - Náhodné číslo určené žadatelem

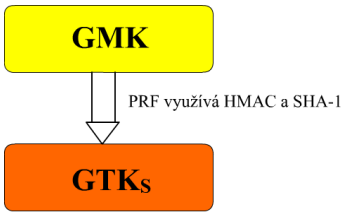


Obr. 4.5: Pairwise hirrarchie

- Náhodné číslo určené ověřovatelem

2. **Group key** Výchozím bodem této hierarchie je klíč GMK (Group Master Key). GMK je vlastně náhodné číslo. Pomocí pseudonáhodné funkce se z GMK a dalších parametrů vytvoří klíč GTK (Group Temporary Key) – viz Obr4.6. Parametry pro vytvoření GTK jsou:

   - MAC adresa ověřovatele (Access Point)
   - Náhodné číslo určené ověřovatelem



Obr. 4.6: Group key hierarchie

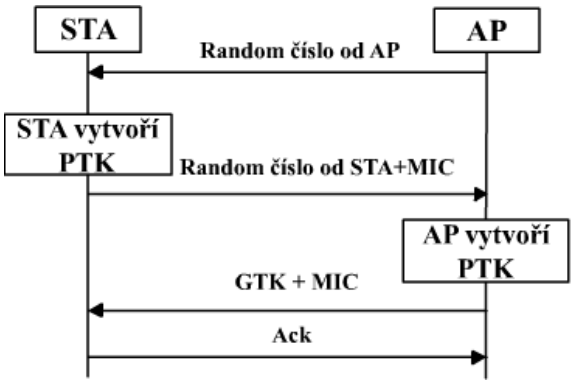
4.4.4 Výměna klíčů

V rámci IEEE 802.11i jsou definovány dvě základní metody výměny klíčů. 4-way handshake pro unicast přenosy a group key handshake pro multicast a broadcast přenosy.

1. **4-way handshake**
   Tato metoda slouží pro sestavení PTK klíče a vyžaduje zaslání 4 paketů mezi žadatelem a ověřovatelem (viz Obr 4.7).

   Postup sestavení PTK:

   (a) Ověřovatel (AP) zašle klientovi náhodné číslo (ANonce).



Obr. 4.7: 4-way handshake (7TA = stanice, AP = access point)

(b) Klient vygeneruje své náhodné číslo. Tím je schopen z MAC adresy AP, zaslaného náhodného čísla a jím vygenerovaného čísla sestavit PTK klíč. Klient zašle své vygenerované číslo a také seznam bezpečnostních parametrů zjištěných na počátku žádosti o připojení do sítě. Celou zprávu opatří kontrolním součtem (MIC – Message Integration Code) za použití KCK klíče. AP tak může zjistit, zda jsou informace a připojené bezpečnostní parametry validní pomocí SNonce a MIC.

(c) AP zašle klientovi bezpečnostní parametry, které normálně vysílá v beacon rámcích nebo probe response paketech. Zároveň pošle GTK klíč šifrovaný KEK klíčem a celou zprávu opatří kontrolním součtem, aby si klient mohl ověřit validitu zaslaných informací a parametrů (GTK + MIC).

(d) Potvrzovací zpráva indikuje, že dočasné klíče byly korektně sestaveny a jsou připraveny k použití v rámci bezpečnostních protokolů (ACK).

2. **Group key handshake** Tato metoda slouží pro aktualizaci GTK klíče. Dříve než je možné tuto metodu použít, je nutné provést výměnu klíčů pomocí 4-way handshake. Nicméně v rámci 4-way handshake je tato metoda interně použita (viz krok 3). Postup při výměně EAPOL-key je shodný s koncovou sekvencí předchozí metody:

(a) AP pošle klientovi GTK klíč šifrovaný KEK klíčem a celou zprávu opatří integritním kódem.

(b) Potvrzovací zpráva indikuje, že dočasné klíče byly korektně sestaveny a jsou připraveny k použití v rámci bezpečnostních protokolů.

5 TLS

Protokol Transport Layer Security (TLS) a jeho předchůdce, Secure Sockets Layer (SSL), jsou kryptografické protokoly, poskytující velmi širokou možnost zabezpečené komunikace po TCP pro služby jako www, elektronická pošta, internetový fax a další datové přenosy, pracující na principu spojení koncových bodů skrz síť. TLS protokol je součástí IETF standardu, naposledy změněném v RFC 5246 v srpnu 2008. TLS vychází z dřívější SSL specifikace vyvinuté společností Netscape. Mezi protokoly SSL 3.0 a TLS 1.0 (nyní jsou dostupné již ve verzi SSL v 3.1 a TLS v 1.2) jsou drobné rozdíly, ale v zásadě pracují stejně. Bohužel i přes svoji příbuznost a blízkost jsou vzájemně nekompatibilní.

TLS využívá pro samotný přenos dat protokol TCP nebo UDP. Zajišťuje důvěrnost a autentičnost přenášených dat i autentizaci komunikujících stran. Z kryptografického hlediska se jedná o hybridní protokol, tj. používá symetrický i asymetrický kryptosystém.

5.1 Historie

SSL protokol byl původně vynalezen společností Netscape. Verze 1.0 vyšla v roce 1994 a nebyla nikdy publikována, verze 2.0 byla zveřejněna v listopadu roku 1994, ale obsahovala několik bezpečnostních děr, a proto se navrhlo SSL v 3.0, které bylo zveřejněna v roce 1996. Tato poslední verze sloužila jako základ pro TLS v1.0, jenž uveřejnila Internet Engineering Task Force(IETF) poprvé v RFC 2246 v lednu 1999 (pracovala na něm v letech 1997-1999). Visa, MasterCard, American Express a mnoho dalších hlavních finančních institutů schválily SSL a TLS pro bezpečné bankovnictví přes Internet.

TLS podléhá trendům. Jeho rozšiřující součásti jsou vytvořené tak, aby podporovaly jak následné, tak i předchozí verze, a mohly se tak různé verze dohodnout.

5.2 Popis

Protokol TLS umožňuje aplikacím komunikovat po síti způsobem, který zabraňuje odposlouchávání či falšování zpráv. Pomocí kryptografie poskytuje TLS svým koncovým bodům autentizaci a soukromí při komunikaci Internetem. Typicky je autentizován pouze server (tedy jeho totožnost je zaručena), zatímco klient zůstává neautentizován. To znamená, že koncový uživatel (ať člověk či aplikace, jako třeba

webový prohlížeč) si může být jist s kým komunikuje. Další úroveň zabezpečení – při níž oba konce „konverzace“ mají jistotu s kým komunikují – je označována jako vzájemná autentizace. Vzájemná autentizace vyžaduje nasazení infrastruktury veřejných klíčů (PKI) pro klienty. Třetí možností komunikace je neautorizované spojení, kdy se pouze zajišťuje šifrovaný přenos a tím dojde k zabránění odposlechu či falšování zpráv.

Velkou výhodou TLS je také jeho plně transparentní činnost pro uživatele, jelikož je TLS součástí aplikační vrstvy ISO modelu. Z hlediska modelu OSI TLS lze umístit do relační a prezentační vrstvy.

VRSTVA	ZABEZPEČENÍ	
Aplikační	PGP, SSH	
Prezentační	TLS	Record protocol
Relační		Handshake protocol
Transportní	-	
Síťová	IPSec	
Datová	bezdrátové spoje (zpravidla IEEE 802.11 a 16)	
Fyzická	kabelové spoje (zpravidla šifrátory ISDN nebo E1)	

Obr. 5.1: ISO model s použitým zabezpečením

TLS přebírá data od aplikace, v rámci prezentační vrstvy je rozdělí na segmenty, které zašifruje. Před přenosem v rámci relační vrstvy naváže spojení, provede vzájemnou autentizaci stanic a vybuduje šifrovaný kanál. Poté využívá služeb transportní vrstvy k samotnému přenosu dat.

Zjednodušeně TLS pracuje na dvouch vrstvách:

- **TLS Rekord Protokol** – je postaven nejníže z daných protokolů TLS a má na starosti, že spojení zůstává bezpečné za pomocí symetrického šifrování. Tento protokol je také používán k zapouzdření všech ostatních TLS protokolů.
- **TLS Handshake Protokol** – umožňuje autentifikaci mezi serverem a klientem, dohodnutí použitých šifrovacích algoritmů a klíčů, dříve než aplikační protokol přenese jakákoliv data.

TLS zahrnuje tři základní fáze:

- dohodu účastníků na podporovaných algoritmech – Handshake protokol (viz. kapitola 5.3)

- výměnu klíčů založenou na šifrování s veřejným klíčem a autentizaci vycházející z certifikátů - ChangChipherSpec protokol (viz. kapitola 5.5)
- šifrování provozu symetrickou šifrou – Report protokol (viz. kapitola 5.4)

Přehled využívaných šifrovacích algoritmů v TLS:

- pro kryptografii s veřejným klíčem: RSA, Diffie-Hellman, DSA
- pro výměnu klíčů: RSA, Diffie-Hellman, ECDH, SRP, PSK
- pro autorizaci: RSA, DSA, ECDSA
- pro symetrické šifrování: RC4, Triple DES, AES, IDEA, DES
- pro hašování: HMAC-MD5, HMAC-SHA

Protokol TLS je založen na výměně záznamů. Každý záznam může být volitelně komprimován, může k němu být připojen autentizační kód MAC a může být zašifrován. Každému záznamu je přiřazen typ obsahu, který určuje protokol vyšší úrovně. Při zahájení spojení vrstva záznamů obaluje jiný protokol – iniciační protokol (handshake protocol), jehož typ obsahu má hodnotu 22.

5.3 Handshake protokol

Během první fáze se klient a server dohodnou na používaných šifrách, výměně klíčů a autorizačních algoritmech a také na Message authentication codes (MACs). Výměna klíčů a autorizovacích algoritmů jsou typicky kryptografie s veřejnými klíči. MACs je vytvořen pomocí hašovacích funkcí využívající HMAC konstrukci z TLS. Získané a odvozené klíče jsou pak použity i pro TLS Record protokol.

Typická inicializace probíhá následovně:

1. Klient pošle zprávu ClientHello oznamující nejvyšší verzi TLS, kterou podporuje, náhodné číslo a seznam doporučených šifrovacích sad a kompresních metod.
2. Server odpoví zprávou ServerHello obsahující zvolenou verzi protokolu, náhodné číslo, šifrovací a kompresní metodu vybranou z klientem nabídnutého seznamu.
3. Server pošle svůj certifikát (Certificate), pokud to zvolená šifra umožňuje.

4. Server může pomocí CertificateRequest vyžadovat certifikát od klienta, aby bylo spojení autentizováno vzájemně.

5. Server pošle zprávu ServerHelloDone, která signalizuje, že ukončil iniciační dohodu na používaných mechanismech.

6. Klient odpoví zprávou ClientKeyExchange, jež může obsahovat Pre-Master-Secret, veřejný klíč nebo nic (v závislosti na zvolené šifře).

7. Klient a server následně z náhodných čísel a Pre-Master-Secret pomocí pečlivě navržené pseudonáhodné funkce vypočítají master secret. Veškeré ostatní klíče jsou odvozeny z něj (a z generovaných náhodných hodnot).

8. Klient nyní odešle zprávu ChangeCipherSpec, jíž v podstatě sděluje „veškerá další data ode mne budou šifrována“. Za pozornost stojí, že ChangeCipherSpec je sám o sobě protokolem záznamové vrstvy s typem 20, nikoli 22.

9. Na závěr klient pošle šifrovanou zprávu Finished obsahující hash a MAC předchozích iniciačních zpráv.

10. Server se pokusí dešifrovat klientovu zprávu Finished a ověřit její hash a MAC. Pokud dešifrování či ověření selže, inicializace je považována za neúspěšnou a spojení by mělo být ukončeno.

11. Konečně server pošle zprávy ChangeCipherSpec a svou zašifrovanou Finished a klient provede analogické dešifrování a ověření.

12. V tomto okamžiku je inicializace dokončena a je povolen aplikační protokol, jehož typem obsahu je 23. Aplikační zprávy vyměňované mezi klientem a serverem budou zašifrovány.

Pomozme si v lepším pochopení typického TLS/SSL spojení tím, co bývá obvykle označováno za šestikrokový TLS proces. TLS naváže stavové spojení dohodnuté výše popsanou iniciační procedurou mezi klientem a serverem. Během inicializace si oba vymění specifikace pro šifrování, které budou při komunikaci používat.

- Inicializace (handshake) začíná když se klient připojí k serveru používajícímu TLS/SSL a požaduje, aby mu server poslal svou identifikaci.

- Server pošle identifikaci v zabezpečené podobě digitálního certifikátu. Certifikát obsahuje jméno serveru, důvěryhodnou certifikační autoritu (CA) a veřejný klíč serveru.

Prohlížeč může kontaktovat důvěryhodnou CA a ověřit pravost certifikátu, než bude pokračovat. Následně prohlížeč nabídne seznam šifrovacích algoritmů a hashovacích funkcí.

- Z tohoto seznamu server vybere nejsilnější šifrování, které také podporuje, a oznámí toto rozhodnutí klientovi.

K vygenerování klíčů seance použitých pro zabezpečení spojení použije prohlížeč veřejný klíč serveru z certifikátu. Zašifruje jím náhodné číslo a zašle je serveru.

- Tato data klient dokáže zašifrovat, ale pouze server je umí rozšifrovat (pomocí svého soukromého klíče): tímto způsobem zůstane klíč skryt před případným odposlouchávajícím, znají jej pouze server a klient.
- Server odpoví dalšími náhodnými daty (která není třeba šifrovat)
- Následně obě strany použijí vybranou hashovací funkci na náhodná data k vytvoření klíče seance.

Tím končí handshake a začíná zabezpečené spojení, které je šifrováno a dešifrováno klíči seance po zbytek svého trvání.

Pokud je libovolný z těchto kroků neúspěšný, selže TLS/SSL handshake a nedojde k vytvoření spojení.

Odvození dalších tajných parametru:
(*Používá se kombinace hašovacích funkcí MD5 a SHA-1*)
Z hodnoty PMS se odvodí MS (Master Secret):
$MS =$
$MD5(PMS \| SHA('A' \| PMS \| CN \| SN)) \|$
$MD5(PMS \| SHA('BB' \| PMS \| CN \| SN)) \|$
$MD5(PMS \| SHA('CCC' \| PMS \| CN \| SN))$.

z hodnoty MS se nakonec generuje blok klíčů BK:
$BK =$
$MD5(MS \| SHA('A' \| MS \| CN \| SN)) \|$
$MD5(MS \| SHA('BB' \| MS \| CN \| SN)) \|$
$MD5(MS \| SHA('CCC' \| MS \| CN \| SN)) \|$
a dále podle potřeby.

Paket Handshake protokolu:

+	bity 0–7	8-15	16-23	24–31
0	22	Version (MSB)	Version (LSB)	Length (MSB)
32	Length (LSB)	Message type	Message length	
64	Message length (cont.)	Handshake message		
...	Handshake message	Message type	Message length	
...	Message length	Handhshake message		

Obr. 5.2: Handshake protokol

Message type - Identifikuje typ zprávy. Dostupné typy jsou:
0 HelloRequest, 1 ClientHello, 2 ServerHello, 11 Certificate, 12 ServerKeyExchange, 13 CertificateRequest, 14 ServerHelloDone, 15 CertificateVerify, 16 ClientKeyExchange, 20 Finished.

Message length - Jedná se o tříbajtovou položku obsahující délku handshake dat, hlavička se nepočítá.

5.4 Protokol TLS záznamů (Report protokol)

Jak už bylo řečeno Report protokol se stará o šifrovaný přenos veškerých informací za pomocí symetrických šifer. Funguje podobně jak TCP protokol, tzn. zapouzdřuje další komunikaci.

Zpracování zprávy:
Z hierarchicky vyšší OSI vrstvy (např. od HTTP) je prevzata zpráva Z.

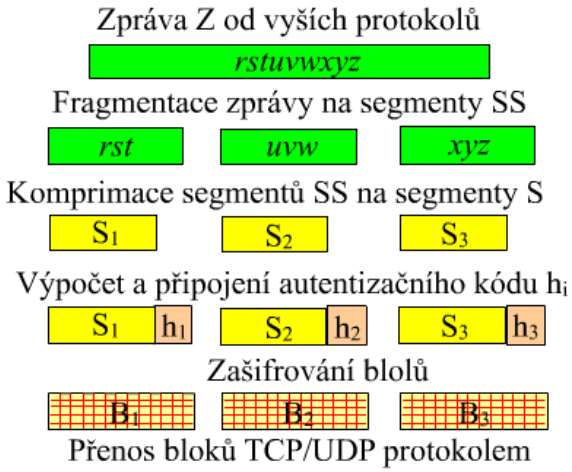
1. fragmentace zprávy Z na segmenty SS o maximální délce 2^{14} bajtu,
2. bezeztrátová komprimace segmentu SS do podoby S,
3. výpocet autentizacního kódu h segmentu S (viz dále),
4. vytvorení bloku $B = S||h$,
5. zašifrování bloku B.

Zašifrovaný blok je opatřen služebními daty a předán transportní vrstvě k přenosu TCP protokolem. Na přijímací straně se provedou inverzní operace v opačném pořadí.

Výpočet autentizačního kódu h segmentu S:
$h = H(AKO||ipad||H(AKO||opad||SN||PT||SL||S))$, kde :
- AKO = autentizační klíč odesílatele,

- ipad = řetězec bajtu $0x36$,
- SN je číslo zprávy,
- PT je číslo protokolu v nadřízené vrstvě OSI,
- SL je délka segmentu S.

Autentizační kód h umožňuje přijímací straně kontrolu, zda nedošlo při přenosu dat k chybě nebo manipulaci.



Obr. 5.3: Segmentace zprávy v TLS

Paket Record protokolu:

+	bity 0–7	8-15	16-23	24–31
0	Protocol	Version (MSB)	Version (LSB)	Length (MSB)
32	Length (LSB)	Zprávy protokolu		
...	Zprávy protokolu (pokračování)			
...	MAC (volitelně)			

Obr. 5.4: Record protokol

Protocol - Tato položka identifikuje typ protokolu vrstvy záznamů obsažený v tomto záznamu. Dostupné typy protokolů jsou:
Version - Identifikuje hlavní a dílčí verzi SSL pro obsaženou zprávu. Pro zprávu ClientHello se nemusí jednat o nejvyšší verzi podporovanou klientem.

Verze jsou: 0 SSLv3 , 1 TLS 1.0 , 2 TLS 1.1

Length - Délka zpráv protokolu. Nesmí překročit 2^{14} bajtů.

Druh protokolu		
HEX	**DEC**	**DRUH**
$0x14$	20	ChangeCipherSpec
$0x15$	21	Alert
$0x16$	22	Handshake
$0x17$	23	Application

Tab. 5.1: Dostupné typy protokolů

Zprávy protokolu - Jedna nebo více zpráv identifikovaných položkou Protocol. Tato položka může být zašifrována v závislosti na stavu spojení.

MAC - Kód ověřující autentičnost zprávy vypočítaný ze zpráv protokolu doplněný o klíče. Položka může být zašifrována nebo může chybět, opět v závislosti na stavu spojení.

5.5 Protokol ChangeCipherSpec

+	bity 0–7	8-15	16-23	24–31
0	20	Version (MSB)	Version (LSB)	0
32	1	1 (CCS protocol type)		

Obr. 5.5: Change Chipher Specifikation protokol

5.6 Protokol Alert

+	bity 0–7	8-15	16-23	24–31
0	21	Version (MSB)	Version (LSB)	0
32	2	Level	Description	

Obr. 5.6: Alert protokol

Level - Položka identifikující úroveň výstrahy. Úrovně jsou:

- Varování - spojení nebo bezpečnost mohou být nestabilní
- Fatální - spojení nebo bezpečnost mohou být kompromitovány nebo došlo k nenapravitelné chybě

Description - Identifikuje typ zasílané výstrahy.
Dostupné popisy jsou:
0 Close notify, 10 Unexpected message (fatal), 20 Bad record MAC (fatal), 21 Decryption failed (fatal), 22 Record overflow (fatal), 30 Decompression failure (fatal), 40 Handshake failure (fatal), 42 Bad certificate, 43 Unsupported certificate, 44 Certificate revoked, 45 Certificate expired, 46 Certificate unknown, 47 Illegal parameter (fatal), 48 Unknown CA (fatal), 49 Access denied (fatal), 50 Decode error (fatal), 51 Decrypt error, 60 Export restriction (fatal), 70 Protocol version (fatal), 71 Insufficient security (fatal), 80 Internal error (fatal), 90 User cancelled (fatal), 100 No renegotiation (warning).

5.7 Bezpečnost

TLS zahrnuje řadu bezpečnostních opatření:

- Klient používá veřejný klíč certifikační autority (CA) k ověření jejího digitálního podpisu v serverovém certifikátu. Lze-li digitální podpis CA ověřit, klient přijme serverový certifikát jako platný certifikát vydaný důvěryhodnou CA.
- Klient ověřuje, zda je vydávající certifikační autorita na seznamu důvěryhodných CA.
- Klient kontroluje dobu životnosti serverového certifikátu. Autentizační proces se zastaví, pokud doba jeho platnosti vypršela.
- K ochraně před útoky typu *Man in the Middle* porovnává klient aktuální DNS jméno serveru se jménem z certifikátu.
- Ochrana před několika známými útoky (včetně Man in the Middle), jako je snaha o použití nižší (méně bezpečné) verze protokolu nebo slabšího šifrovacího algoritmu.
- Opatření všech aplikačních záznamů pořadovými čísly a používání těchto čísel v MAC.
- Používání ověřovacího kódu zprávy rozšířeného o klíč, takže jen vlastník klíče dokáže MAC ověřit. Definováno v RFC 2104.
- Zpráva ukončující inicializaci (*Finished*) obsahuje hash všech zpráv vyměněných v rámci inicializace oběma stranami.

- Pseudonáhodná funkce rozděluje vstupní data na poloviny a zpracovává každou z nich jiným hashovacím algoritmem (MD5 a SHA-1), pak je XORuje dohromady. To poskytuje ochranu, pokud by byla nalezena slabina jednoho z algoritmů.

6 GSM

Bezpečnost sítě GSM je postavena na čtyřech základních bodech.

- Autentifikace uživatele
- Utajení identity uživatele
- Utajení signalizace
- Utajení přenášených dat

Zajištění zabezpečení sítě má na starost několik jejích částí. Konkrétně se jedná o Subsciber Identity Module (SIM), mobilní stanici (MS) a autentifikační centrum sítě (AUC), které je součástí provozního a servisního centra sítě (OMS).

SIM obsahuje mimo jiné International Mobile Subsciber Identity (IMSI), tajný klíč autentifikace Ki, dále algoritmus pro generování šifrovacího klíče A8, autentifikační algoritmus A3 a osobní identifikační číslo PIN.

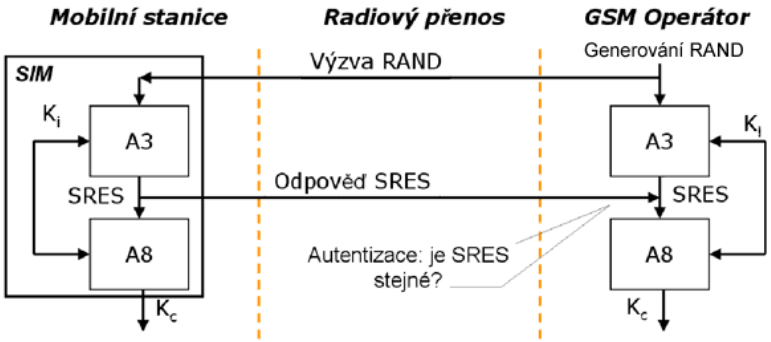
Mobilní stanice obsahuje šifrovací algoritmus A5.

AUC obsahuje databázi s identifikačními a autentifikačními údaji uživatelů sítě. V databázi jsou uloženy IMSI, TMSI (Temporary Mobile Subscriber Identity), LAI (Local Area Identity) a tajný klíč Ki, který je unikátní pro každého uživatele.

Pro provedení autentifikace uživatele je třeba součinnosti všech tří výše uvedených částí. To zvyšuje bezpečnost sítě a zlepšuje tak ochranu např. před „duplikováním“ SIM.

6.1 Autentifikace uživatele

Autentifikace uživatele je prováděna při každém vstupu uživatele do sítě. GSM síť ověřuje identitu uživatele pomocí mechanismu challenge-response. Síť pošle MS 128-bitové náhodné číslo, které označujeme jako RAND. V MS je číslo RAND předáno modulu SIM, který na základě znalosti Ki a algoritmu A3 vypočítá odpověd, kterou označíme jako SRES nebo K3. SRES je počítáno na základě čísla RAND, tajného klíče Ki a algoritmu A3, tzn. SRES=A3(Ki,Rand) a má 32 bitů. MS po vypočtení SRES pošle výslednou hodnotu zpátky do sítě. Síť po přijetí SRES vypočítaný v SIM provede stejný výpočet jako byl proveden v SIM. K výpočtu využije opět algoritmus A3, kód Ki a čísla RAND, které bylo zasláno do MS. Pokud síť dojde



Obr. 6.1: Autentifikace v GSM

ke stejnému výsledku jako obdržela od MS, je autentifikace úspěšná. Pokud ne, je spojení ukončeno. Schéma autentifikace uživatele sítě je na obrázku 6.1.

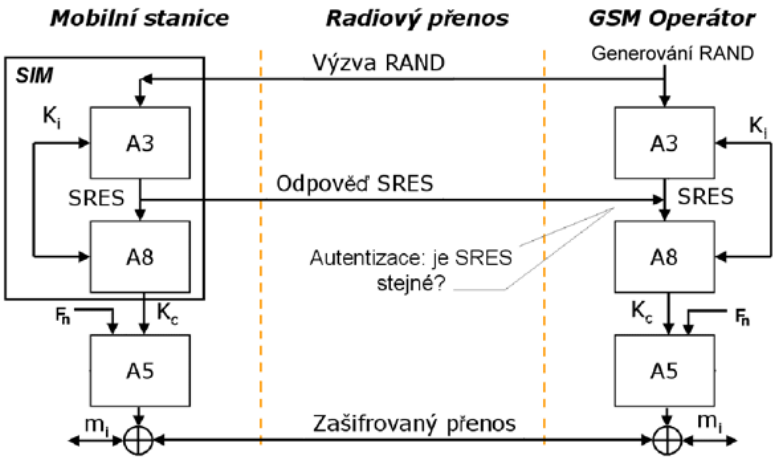
Zde je dobré si povšimnou, že Ki se sítí vůbec nepřenáší. Hodnota Ki je totiž napevno uložena jak v uživatelově SIM, tak v AUC, a přenos tohoto klíče tak není nutný. Dalším prvkem posilujícím zabezpečení je omezení manipulace s Ki v rámci SIM. Jediné, co SIM dovolí s Ki provádět, jsou operace kódování A38. Kromě těchto operací je přístup ke Ki zamezen, takže tento kód nelze jen tak z SIM karty přečíst.

6.2 Zabezpečení přenášených dat

Kromě autentifikace uživatele je třeba zajistit i bezpečnost přenášených dat, hovoru a signalizace. To je uskutečněno pomocí algoritmu A5. Jako klíč pro tento algoritmus se přitom používá hodnota K8. To je hodnota, která je získána již při autentifikaci uživatele, kdy je vypočítána obdobným způsobem jako K3. Rozdíl je pouze ten, že K3 se počítá pomocí algoritmu A3, zatímco K8 pomocí algoritmu A8, tzn. $K8 = A8(Ki, Rand)$ a má 64 bitů. Po vypočtení K8 je tato hodnota uložena do SIM a při zahájení komunikace je použita jako klíč pro algoritmus A5. Dalším vstupem A5 je Fn, což je 22-bitové číslo rámce, tzn. že celkový zápis vypadá takto, $Hi = A5(K8, Fn)$ a je 114 bitů dlouhé.

A5 je proudová šifra a v GSM se využívá několik jejich variant a to konkrétně :

- A5/1 bezpečnější verze využívaná v Americe a Evropě
- A5/2 slabší verze A5/1 využívaná ostatními zeměmi
- A5/3 šifrovací algoritmus používaný v UMTS (Katsumi)



Obr. 6.2: Šifrování dat v GSM

Obě operace A3 a A8 bývají často implementovány jako jedna, kterou označujeme jako A38, nebo také COMP 128.

Délka klíče [b]	32	40	56	64	128
Potřebný čas	42s	3, 05h	23 let	5849 let	$10,8 \cdot 10^{22}$ let

Tab. 6.1: Doba potřebná k provedení brute-force útoku na klíče různých délek počítačem s rychlostí 100 000 000 operací za sekundu.

Délka klíče [b]	Potřebný počet počítačů			
	1h	1 den	1 týden	1 měsíc
40	3	1	1	1
56	200159	8340	1191	39
64	$51,2 \cdot 10^6$	$21,3 \cdot 10^5$	305006	1016
128	$9,4 \cdot 10^26$	$3,94 \cdot 10^25$	$5,63 \cdot 10^24$	$1,88 \cdot 10^23$

Tab. 6.2: Počet počítačů s rychlostí 100000000 operací za sekundu potřebných k provedení brute-force útoku na klíče různých délek v určitém čase.

6.3 Zabezpečení a útoky v praxi

Jak z výše nastíněných principů vyplývá, veškerá komunikace v síti GSM je šifrována a uživatelé jsou pokaždé podrobeni autentifikaci. Z tabulky 16.1 vidíme, že použití primitivního brute-force útoku není prakticky možné, protože klíče použité v GSM jsou příliš dlouhé. Tabulka 6.2 pak navíc ukazuje, že řešením není ani použití více

počítačů. I když jejich výpočetní schopnosti rostou a ceny klesají, stále by bylo vybudování sítě pro odposlechy velmi nákladnou investicí. Případný útok tedy musí být proveden jinak.

Z nastíněných principů je zřejmé, že veškeré zabezpečení by bylo zbytečné, pokud by se útočníkovi podařilo dostat k tajnému klíči Ki. Pak by již nebyl žádný problém dopočítat hodnoty K3, K8 a duplikovat tak SIM-kartu, či odposlouchávat hovor napadeného uživatele.

Jak jsme již uvedli výše, hodnota Ki je uložena v SIM a AUC. Dostat klíč z AUC je prakticky vyloučeno, pro případný útok tak zbývá varianta získání klíče přímo ze SIM nebo z komunikace mezi MS a sítí. Obě zmíněné varianty s sebou přinášejí problémy. Přesto lze tento klíč získat a realizovat tak útok na GSM síť.

6.3.1 Získání Ki ze SIM

Využití kryptografické analýzy

Jedním z možných útoků je získání Ki přímo ze SIM karty proti které útočíme. Je známo, že z karty se Ki nedá přečíst, můžeme se ale pokusit Ki zjistit tak, že kartě předkládáme různé hodnoty RAND a z jejích reakcí pak zjistíme Ki. Z toho plyne, že pokud chceme tento útok realizovat, je zapotřebí mít příslušnou kartu fyzicky k dispozici. Dále budeme potřebovat speciální zařízení, které bude do SIM karty zasílat různé RAND a následně přijímat a analyzovat odezvu SIM. Po vznesení cca 180 000 dotazů (RAND kódu) je možné získat hodnotu Ki uloženého v příslušné SIM kartě.

Útok probíhá hledáním kolizí mezi vrácenými hodnotami SRES. Postupně zadáváme dvojice RAND tak, aby byly „skoro stejné“. Měníme tedy pouze jeden nebo dva byty zadávaného čísla, a to tak, že při hledání 0. a 8. bytu klíče Ki měníme pouze 0. a 8. byte RAND, při hledání 1. a 9. měníme 1. a 9. byte RAND, atd. Získáme-li pro jednu kombinaci dvou RANDu stejný výsledek SRES, můžeme na základě znalosti algoritmu funkce A38 dopočítat příslušné dva byty klíče Ki. Při luštění je třeba dát pozor na falešné kolize, které mohou vznikat kvůli nastavení posledních deseti bitu na nulu.

Tento útok trvá podle dostupných informací cca 8 hodin. Vzhledem k tomu, že limitujícím faktorem zde je rychlost s jakou umí analyzující zařízení komunikovat se SIM kartou a jak rychle umí výsledné hodnoty analyzovat, lze se domnívat, že tento čas lze dále zkrátit. Pokud má tedy útočník atakovanou kartu k dispozici po tuto dobu, není pro něj velkým problémem získat Ki dané karty a následně ji buď duplikovat nebo odposlouchávat přenášená data.

Je třeba dodat, že nové karty jsou proti tomuto typu útoku celkem dobře chráněny. Metody jsou různé, často se používá umělé zpomalení karty, které odezvy karty brzdí tak, že je tento způsob útoku nemožný. Další možností je nastavení počtu operací (provedení algoritmu A38), které je možno s kartou provést. Při tomto typu útoku tak dojde ke zničení karty.

Postranní kanály
IBM založilo v roce 2002 divizi, která se naplno věnuje technikám umožňujícím využít nedostatky některých mobilních technologií. Nedostatky, které umožní zcela cizím osobám klonovat libovolnou SIM kartu.

Vymyšlenou novinkou je v tomto případě usnadnění procesu vedoucího k prolomení ochranných prvků SIM karet. Kdokoliv s minimem schopností, ale vlastnící potřebné zařízení (čtečka smart karet, PC vybavené potřebným softwarem a SIM karta), může použít cizí SIM kartu a vytvořit během několika minut klon.

Princip funkčnosti je založen na částicovém či štěpném útoku (*partitioning attack*), a tím se dosáhne rapidního zkrácení času - místo zdlouhavého louskání potřebného tajného kódu dojde k monitorování postranních kanálů - spotřeby elektrické energie a elektromagnetického vyzařování. COMP128 klíč z SIM karty je potom možné získat, aniž by bylo nutné absolvovat dlouhé zkoušení. Podle informací IBM přitom postačí, aby SIM karta provedla sedmkrát pokus o vyhodnocení klíče s neznámým klíčem.

IBM zároveň uvádí, že vyvinuli metodu, vedoucí k ochraně tohoto druhu útoku. Ta spočívá v rozmělnění doposud jednoduchých vyhledávacích dotazů do tabulek v paměti mobilního telefonu do série doplněné o náhodné dotazy. Postranní únik informací je potom pro případného hackera nepoužitelný.

6.3.2 Získání Ki odposlechem hovoru

S pomocí dále popsané procedury se síť GSM snaží zabránit odposlechu hovorů přenášených vzduchem mezi mobilním telefonem a sítí GSM. K tomu slouží proudové šifrovací schéma A5, které využívá dočasný klíč K8, dohodnutý během poslední autentizační fáze.

Data přenášená od MS směrem do sítě a data jdoucí opačným směrem procházejí různými kanály. Data jsou organizována po paketech, v kanálech se seskupují do úseků po 114 bitech a jsou vysílána po doplnění o synchronizační údaje v tzv. *burstech*. Tato organizace je zavedena proto, že GSM používá metodu časového sdílení jednoho kanálu (TDMA – Time Division Multiple Access), která v jednom TDMA

rámci vyhrazuje osm časových slotů. V každém z nich přitom může probíhat jiná komunikace.

Právě číslo rámce TDMA, v jehož časovém slotu je daný burst přenášen, se spolu s K8 podílí na generovaní hesla algoritmem A5 ($Hi = A5(K8, Fn)$). Číslo rámce je pro útočníka provádějícího pasivní odposlech známé.

Algoritmus A5 není z důvodu přenosové rychlosti implementován v SIM kartě, ale přímo v MS. Po algoritmu se požaduje, aby byl během doby 4,615 ms (trvání rámce) schopen vygenerovat 228 bitů hesla, protože oba kanály (odchod dat a příjem dat) musí použít různá hesla.

Tím se nám nabízí další možnosti, jak lze Ki získat, tedy odposlech hovoru. U algoritmu A5, se využívají dvě varianty. Silnější variantu A5/1 a slabší verzi A5/2. Útok na slabší verzi algoritmu popsali např. Ian Goldberg a David Wagner z University of California v Berkeley. Pomocí jimi popsaného útoku je možné tento slabší algoritmus prolomit v řádu milisekund.

Zajímavější je situace u silnějšího algoritmu A5/1, který se používá i u nás. Problém A5/1 spočívá v tom, že dohodnutý klíč není ve skutečnosti 64-bitový, jak se tvrdí, ale pouze 54-bitový, a na 64-bitový je rozšířen nulami zprava. Toto zúžení klíčového prostoru spolu s dalšími drobnými chybami v návrhu A5/1 je natolik drastickým opatřením, že již dříve bylo ukázáno, že tento algoritmus je teoreticky možné prolomit. Útok byl však časově náročný (zapotřebí bylo něco mezi 2^{40} až 2^{45} výpočetních kroků) a navíc vyžadoval zachycení třiceti minutové komunikace, což je pro praktické využití nereálné.

Ve studii profesora A. Shamira a A. Biryukova je popsán útok, který lze realizovat na dnes dostupném počítači (128 MB RAM a dva pevné disky každý o kapacitě 73 GB), nutné je zachytit první dvě minuty hovoru, ve studii popsanou analýzou pak lze nalézt klíč relace za méně než 1 vteřinu. Protože GSM telefony vysílají frame každých 4.6 milisekundy, znamená to, že dvě minuty konverzace obsahují $(120 * 1000)/4.6$, tedy méně než 2^{15} framů. Počet nutných kroků k nalezení klíče je pak mezi 2^{37} až 2^{48} . Útok byl verifikován na aktuální implementaci algoritmu A5/1. Studii A. Shamira a A. Biryukova lze nalézt na http://cryptome.org/a5.ps .

Druhá varianta tohoto útoku podle studie vystačí s odposlechnutím úvodních dvou sekund konverzace, přičemž výpočet se prodlouží na několik minut.

7 VÝSLEDKY PRÁCE

Jako součást této diplomové práce vznikl software za účelem výuky výše popsané problematiky. Vzhledem k obsahu, rozsahu a struktuře výukové části jsem jako sdělovací prostředí zvolil internetovou prezentaci, nebo-li internetové stránky, s ohledem na požadavek zadání diplomové práce, že daný software má být spustitelný na běžném webovém prohlížeči. Díky tomu odpadá i problém uniplatformnosti.

Přínosem této aplikace je demonstrace kryptografických protokolů v dnes nejpoužívanější technologii sdělování informací - www. Další nemalou výhodou je fakt, že výuka bude dostupná kdekoli s připojením na internet. Lokální spuštění stránek je popsáno v kapitole 7.4.2.

7.1 Využité programovací jazyky

7.1.1 PHP

Dynamické stránky, tj. stránky nejprve vygenerované serverem a poté odeslané klientovi, jsou v současné době nezbytnou součástí každé složitější internetové prezentace. K hlavním skriptovacím jazykům, které se používají k tvorbě těchto stránek, patří ASP (Active Server Pages) a PHP. Právě jazyk PHP jsem si zvolil pro svoji tvorbu. Jedná se o validní jazyk podle všech standardů.

Server může PHP skripty teoreticky hledat ve všech odesílaných souborech, ale zpravidla je nakonfigurován tak, aby je hledal v souborech s příponami *.php*, *.php*3 nebo *.phtml*. Příkazy PHP jsou vkládány přímo do HTML kódu a jsou od něj odděleny tagy <?*php*? >. [20]

PHP (PHP: Hypertextový preprocesor, původně Personal Home Page) je skriptovací programovací jazyk, určený především pro programování dynamických internetových stránek. Nejčastěji se začleňuje přímo do struktury jazyka HTML, XHTML či WML, což lze využít při tvorbě webových aplikací.

PHP skripty jsou většinou prováděny na straně serveru, tzn. že každá stránka, která obsahuje PHP skripty, server nejprve vezme a vykoná všechny příkazy v PHP, které jsou ve stránce uvedené, poté pošle klientovi již čistý HTML kód, který je výsledkem běhu skriptu. Syntaxe jazyka je inspirována několika programovacími jazyky (Perl, C, Pascal a Java). PHP je nezávislý na platformě (multiplatformnost), skripty fungují bez větších úprav na mnoha různých operačních systémech. Podporuje mnoho knihoven pro různé účely - např. zpracování textu, grafiky, práci se soubory, přístup k většině databázových systémů (mj. MySQL, ODBC, Oracle, PostgreSQL, MSSQL), podporu celé řady internetových protokolů (HTTP, SMTP,

SNMP, FTP, IMAP, POP3, LDAP ...).

Jazyk PHP je dynamicky typový, tzn. že datový typ proměnné se určí v okamžiku přiřazení hodnoty. Pole se dají indexovat číselnými indexy (jako v jazyce C), nebo mohou fungovat jako hash-mapa. Stejné pole může obsahovat oba typy indexů. Pole jsou heterogenní (stejné pole může obsahovat prvky různých typů). Řetězce lze uzavírat do uvozovek nebo do apostrofů. Možnost využití nativních funkcí operačního systému (možná nekompatibilita s jiným OS). PHP je OpenSorce. PHP ale neumožňuje překlad do byte kódu, PHP skript se při každém požadavku překládá znovu.

7.1.2 CSS

CSS (Cascading Style Sheets - tabulky kaskádových stylů) je to jazyk pro popis způsobu zobrazení stránek napsaných v jazycích HTML, XHTML nebo XML. Jazyk byl navržen standardizační organizací W3C. Hlavním smyslem je umožnit oddělit vzhled dokumentu od jeho struktury a obsahu. [22]

7.1.3 JavaScript

JavaScript je multiplatformní, objektově orientovaný skriptovací jazyk. 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.

Program v JavaScriptu se obvykle spouští až po stažení WWW stránky z Internetu (tzv. na straně klienta), na rozdíl od ostatních jiných interpretovaných 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. Na druhé straně JavaScrip nezatěžuje server a požadavky na výpočetní výkon převádí na stranu uživatele.[23]

7.1.4 AJAX

AJAX (Asynchronous JavaScript and XML) je obecné označení pro technologie vývoje interaktivních webových aplikací, které mění obsah svých stránek bez nutnosti jejich znovunačítání. AJAX využívá technologií HTML (nebo XHTML) a CSS pro

prezentaci informací, JavaScript pro zobrazování a dynamické změny prezentovaných informací.

Mezi výhody patří odstranění nutnosti znovunačtení a překreslení celé stránky při každé operaci, které jsou nutné u klasického modelu WWW stránek - při akci se celá stránka musí znovu načíst ze serveru, třebaže se na ní jen například aktualizuje minimum informací a veškerý zbytek obsahu zůstává stejný. Prostřednictvím AJAXu proběhne odesílání provedení akce uživatele na pozadí, server zašle jen ty části stránky, které se změnily, a jen tyto části se uživateli na stránce aktualizují a překreslí. Uživatel tak má pocit mnohem větší plynulosti práce, která se blíží běžným desktopovým aplikacím.

Z toho vyplývá také potenciál snížit zátěž na webové servery a síť obecně. Jelikož není potřeba při každém požadavku sestavit celý HTML dokument, ale pouze provedené změny, je množství vyměňovaných dat výrazně nižší a teoreticky to může mít příznivý vliv i na zátěž serverů.

Mezi nevýhody patří hlavně změny v paradigmatu používání webu: webové stránky se chovají jako plnohodnotná aplikace se složitou vnitřní logikou, nikoli jako posloupnost stránek, mezi kterými se lze navigovat i pomocí tlačítek Zpět a Další. [21]

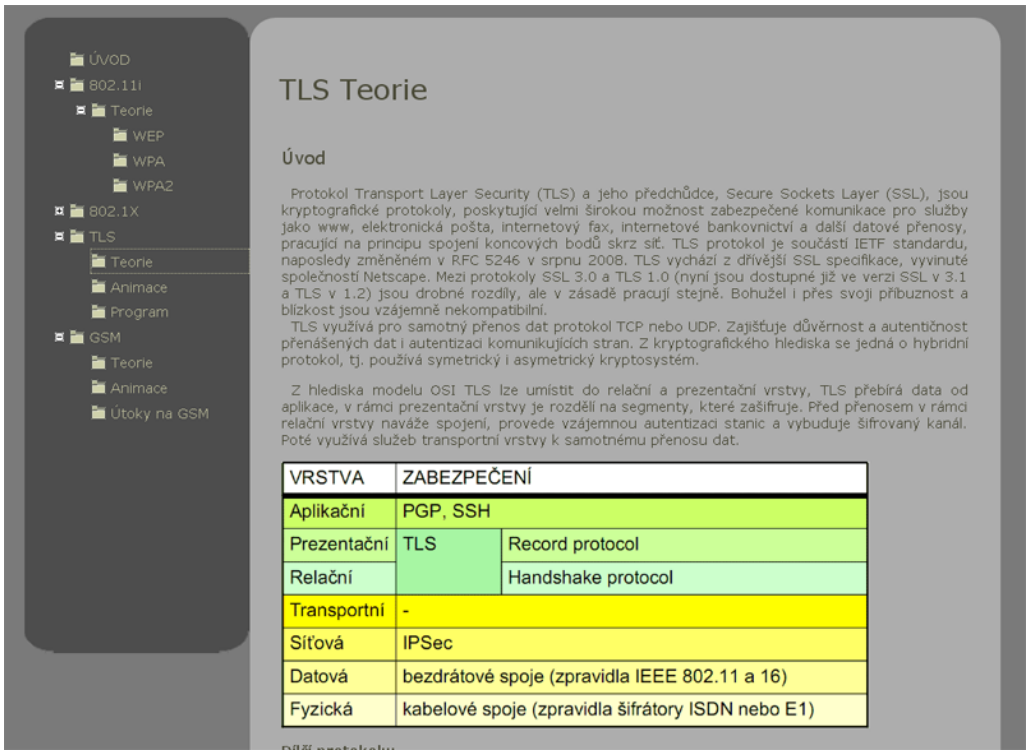
7.2 Vzhled a ovládání programu

Tvář programu bude situovaná na hlavní okno, kde budou veškeré informace a postranní přehledné strukturované menu. Menu je pomocí JavaScriptu plně animované (rolovací), má vlastní jednoduchý kód v souboru *menu.php*. Pomocí css jsou textové položky menu (tagy p,a) nahrazeny grafickými komponenty.

Obr. 7.1: Ukázka grafického strukturovaného menu

Tělo programu je podle vybrané položky menu plněno informacemi získávanými ze serveru za pomocí AJAXu, tzn. že je na server poslán jen dotaz o informace

(stránku) a ta je mu vrácena. Naprogramované funkce poté změní jen potřebnou část stránky bez nutnosti reloaudu.



Obr. 7.2: Náhled aplikace

Co se týká celkového vzhledu, tak jej lze snadno měnit pomocí CSS stylů, které jsou obsažené v *CSS/stranka.css*.

7.3 Animace

Krom textu a ilustrací program také obsahuje animace pro lepší pochopení některých částí výuky.

Animace byly vytvářeny v Adobe Flash CS4 Profesional©, tj. grafický vektorový program. Flash má také vlastní implementovaný programovací jazyk ActionScript, který slouží k rozvinutí všech možností interaktivní animace a vývoji robustních aplikací. V aktuálních verzích je ActionScript poměrně vyspělý objektově orientovaný programovací jazyk.

Flash exportuje soubory do dvou základních formátů, pro nás však je použitelný jen tento:
.swf – v tomto formátu má soubor malou velikost, může být přehrávaný ve webovém prohlížeči, ale k jeho běhu je nutný přehrávač – Adobe Flash Player, který je

volně ke stažení na stránkách společnosti.

Animace mají dvě možnosti přehrávání, buď plynulé přehrávání celé animace s možností pauzy, nebo krokovací režim, kdy se uživateli přehraje jeden krok animace a dále vyčkává na další možnosti uživatele (přejít na následující/předchozí animaci). Mezi oběma styly přehrávání se lze v průběhu animace přepínat. Na konci animace má ještě uživatel možnost přetočit a pustit si animaci znovu. Vše se ovládá tlačítky.

Obr. 7.3: Tlačítka animací

7.4 Požadavky aplikace

Díky tomu, že je aplikace funkční jako www, nejsou na ni kladené žádné velké nároky, je u ni zajištěna multiplatformnost a nezávislost na architektuře. Všechny přídavné části ať už to je PHP, Java, nebo Flash jsou Open Sorce produkty a jsou dostupné na stránkách vydavetelů.

7.4.1 Spuštění na serveru

Jediný požadavek ze strany serveru je, aby na něm byl nainstalován PHP server, který je dnes ale standardně dostupný na většině serverech.

Klient si pak musí :

- Zapnutý JavaScript ve svém webovém prohlížeči.
- Pro přehrání animací je potřeba mít nainstalovaný Adobe Flash Player dostupný na http://get.adobe.com/flashplayer/?promoid=DAFYL
- Pro spuštění Java aplikací potom Java JRE balíček dostupný na : http://java.sun.com/javase/downloads/index.jsp

7.4.2 Lokální spuštění

Aby si uživatel mohl spustit software na svém počítači např bez využití internetu, bude potřebovat výše zmíněné balíčky pro Flash a Javu a navíc nainstalovaný server

PHP.

Postup instalace PHP: Prvně je potřeba stáhnou instalátor : $http://www.slunecnice.cz/sw/php-triad/$
Po instalaci se objeví na nabídce start složka programs - PHPTriad - Apache Console - start apache a tím spustíme Apache. Vyzkoušet funkčnost můžete zadáním této adresy do webového prohlížeče : $http://localhost/$. Pokud se objeví uvítací text, tak vše proběhlo v pořádku.

Druhým krokem je nutné vytvořil libovolný adresář ve složce c:\apache\htdocs\. Do této nově vytvořené složky vložíme stránky a potom už stačí do prohlížeče zadat cestu : $http://localhost/NAZEV slozky/cz/index.php$

8 ZÁVĚR

Textová část této diplomové práce obsahuje podrobnější zpracování principů a funkčností kryptografických a autentizačních protokolů TLS, KERBEROS, IPsec, 802.1X, protokoly obsažené ve standardu 802.11i a autentizace a bezpečnost komunikace v GSM síti. Bylo popsáno jak je v daných protokolech aplikována autentizace stran, integrita dat, šifrování, získání tajných parametrů komunikace, popřípadě jaký vliv mají slabší místa protokolů na bezpečnost. Závěr textové části obsahuje popis programu. Všechny tyto nastudované a získané informace vedly k vytvoření aplikace vhodné pro výuku dané problematiky.

Vzhledem k obsahu, rozsahu, struktuře a maximální dostupnosti výukové části softwaru byl primárně vytvořen jako internetové stránky za použití PHP, JavaScriptu, AJAXu. Tím je zajištěna jak multiplatformnost, tak nezávislost na architektuře. Aplikace může být využívána jako klasické webové stránky, a proto ji stačí umístit na jakýkoliv webový server s podmínkou, že je tam nainstalován i PHP server, nebo může být spuštěna lokálně po instalaci PHP serveru. V obou případech ještě musí být povolen JavaScript.

Kromě textů a ilustrativních částí software poskytuje i interaktivní animace a aplikace vytvořené v Adobe Flash CS4©, na kterých je snáze pochopitelná problematika. K přehrání je potřeba mít nainstalován Adobe Flash Player a Java JRE balíček, volně dostupný na stránkách vydavatelů.

Software je rozdělen na grafickou a obsahovou část, která je rozčleněna na několik souborů podle kapitol, aby se daly obsažené informace snadno upravovat či doplňovat o nové. Stránky byly psány pro Mozila Firefox, nicméně byly testovány na několika ostatních prohlížečích jako Opera, Konqueror, či IE 7 a uspěly validací kódu W3C nejvyšší normy jak pro CSS tak HTML.

LITERATURA

[1] Dierks T.,Rescorla E. *The Transport Layer Security (TLS) Protocol Version 1.2* [online],RTFM, Inc. Poslední aktualizace srpen 2008 [cit. 15. 5. 2009]. Dostupné z URL: <http://tools.ietf.org/html/rfc5246>.

[2] Dierks T.,Rescorla E. *The Transport Layer Security (TLS) Protocol Version 1.1* [online],RTFM, Inc. Poslední aktualizace květen 2006 [cit. 15. 5. 2009]. Dostupné z URL: <http://tools.ietf.org/html/rfc4346>.

[3] Dierks T.,Allen C. *The TLS Protocol Version 1.0* [online],Certicom. Poslední aktualizace Leden 1999 [cit. 15. 5. 2009]. Dostupné z URL: <http://tools.ietf.org/html/rfc2246>.

[4] Medvinsky A.,Hur M. *Addition of Kerberos Cipher Suites to Transport Layer Security (TLS)* [online],CyberSafe Corporation. Poslední aktualizace říjen 1999 [cit. 15. 5. 2009]. Dostupné z URL: <http://tools.ietf.org/html/rfc2712>.

[5] Chown P. *Advanced Encryption Standard (AES) Ciphersuites for Transport Layer Security (TLS)* [online],Skygate Technology. Poslední aktualizace červen 2002 [cit. 15. 5. 2009]. Dostupné z URL: <http://tools.ietf.org/html/rfc2712>.

[6] Santesson S. *TLS Handshake Message for Supplemental Data* [online],VeriSign, Inc. Poslední aktualizace září 2006 [cit. 15. 5. 2009]. Dostupné z URL: <http://tools.ietf.org/html/rfc4680>.

[7] Hollenbeck S. *TLS Handshake Message for Supplemental Data* [online],Microsoft. Poslední aktualizace květen 2004 [cit. 15. 5. 2009]. Dostupné z URL: <http://tools.ietf.org/html/rfc3749>.

[8] Repiquet J. *SSL/TLS Sequence Charts* [online]. Poslední aktualizace 17.6.2008 [cit. 15. 5. 2009]. Dostupné z URL: <http://www.tech-invite.com/Ti-SSL.html>.

[9] Aboba B. a spol *Extensible Authentication Protocol (EAP)* [online]. Poslední aktualizace červen 2004 [cit. 15. 5. 2009]. Dostupné z URL: <http://tools.ietf.org/html/rfc3748>.

[10] Několk autorů *Extensible Authentication Protocol* [online]. Poslední aktualizace 2.5.2009 [cit. 15. 5. 2009]. Dostupné z URL: <http://en.wikipedia.org/wiki/Extensible_Authentication_Protocol EAP-TLS>.

[11] D. Rohleder, V. Lorenc. *802.1X - autentizace v počítačových sítích* Zpravodaj ÚVT MU. ISSN 1212-0901, 2008, roč. XIX, č. 1, s. 2-4. Poslední aktualizace 14.4.2009 [cit. 15. 5. 2009]. Dostupné z URL: <http://www.ics.muni.cz/zpravodaj/articles/590.html>.

[12] Několk autorů *IEEE 802.11i-2004* [online]. Poslední aktualizace 15.5.2009 [cit. 15. 5. 2009]. Dostupné z URL: <http://en.wikipedia.org/wiki/802.11i>.

[13] Halasz D. *IEEE 802.11i and wireless security* [online]. Poslední aktualizace 25.8.2004 [cit. 15. 5. 2009]. Dostupné z URL: <http://www.embedded.com/columns/specialreports/34400002?_requestid=43446>.

[14] Několik autorů *GSM Security* [online]. Poslední aktualizace 15.5.2009 [cit. 15. 5. 2009]. Dostupné z URL: <http://www.gsm-security.net/gsm-security-faq.shtml>.

[15] Margrave D.,George Mason University: *GSM Security and Encryption* [online]. , Poslední aktualizace 7.5.2003 [cit. 15. 5. 2009]. Dostupné z URL: <http://www.hackcanada.com/blackcrawl/cell/gsm/gsm-secur/gsm-secur.html>.

[16] Stallings, W. : *Cryptography and Network Security : principles and practices* 4. vydání, vyd. Upper Saddle River : Pearson Prentice Hall, 2006, ISBN 0-13-187316-4.

[17] Ferguson ,N. : *Practical cryptography* vyd. New York : Wiley, 2003, ISBN 0-471-22357-3.

[18] Doseděl , T.: *Počítačová bezpečnost a ochrana dat* vyd. Brno : Computer Press, 2004, ISBN 80-251-0106-1.

[19] Schneier, B.: *Applied Cryptography* John Wiley, New York 1996.

[20] Několk autorů *PHP* [online]. Poslední aktualizace 14.5.2009 [cit. 15. 5. 2009]. Dostupné z URL: <http://cs.wikipedia.org/wiki/Php>.

[21] Několk autorů *Asynchronous JavaScript and XML* [online]. Poslední aktualizace 16.4.2009 [cit. 15. 5. 2009]. Dostupné z URL: <http://cs.wikipedia.org/wiki/Asynchronous_JavaScript_and_XML>.

[22] Několk autorů *Cascading Style Sheets* [online]. Poslední aktualizace 2.4.2009 [cit. 15. 5. 2009]. Dostupné z URL: <http://cs.wikipedia.org/wiki/Cascading_Style_Sheets>.

[23] Několk autorů *JavaScript* [online]. Poslední aktualizace 3.4.2009 [cit. 15. 5. 2009]. Dostupné z URL: <http://cs.wikipedia.org/wiki/Javascript>.

SEZNAM SYMBOLŮ, VELIČIN A ZKRATEK

AAD Přídavné autentikační data – Additional Authentication Data

AES Rozšířený šifrovací standart – Advanced Encryption Standard

AH Authentication Header – Authentication Header

AJAX Asynchroní JavaScript a XML – Asynchronous JavaScript and XML

AP Přístupový bod – Access point

AS Server autentizace – Authentication Server

ASP Aktivní stránkový server – Active Server Pages

AUC autentifikační centrum sítě – Authentication unix center

CA Certifikační autorita – Certifikate authoority

CBC-MAC Cipher Block Chaining Message Authentication Code

CCMP Counter-Mode/CBC-MAC Protocol

DES standard pro šifrování dat – Data Encryption Standard

DoS Odmítnutí služby – Denial of Service

EAP Rozšířený autentikační protokol – Extensible Authentication Protocol

EAPOL EAP na LAN sítích – EAP over LANs

ESP Encapsulated Security Payload

FIPS Federal Information Processing Standard

GMK Skupinový hlavní klíč – Group Master Key

GTK Skupinový dočasný klíč – Group Temporary Key

HMAC Hash Message Authentication Code

IKE Internet Key Exchange

IV Inicializační vektor – Initialization Vector

IVC Integrity Check Value

IPsec IP security

KCK Potvrzovací klíč – Key Confirmation Key

KDC Cenrtum distribuce klíčů – Key Distribution Center

KEK Šifrovací klíč – Key Encryption Key

MAC Message authentication code

MAC MAC adresa – Media Access Control

MIC Message Integration Code

NAT Překlad síťových adres – Network Address Translation

OCB Offset Codebook

OMS Servisního centra sítě

PAD Bajtová vycpávka

PHP Programovací jazyk – Personal Home Page

PKI Infrastruktura veřejných klíčů – Public Key Infrastructure

PPP Poin-to-Point Protokol

PMK Pairwise Master Key

PMT Pairwise Master Key

PTK Pairwise Transient Key

QoS Quality of Service

SA Bezpečnostní asociace – Security Association

SAD Databáze bezpečnostní asociace – Security Association Database

SIM Subsciber Identity Module

SP Bezpečnostní politika – Security Policy

SPD Databáze bezpečnostní politiky – Security Policy Database

SPI Security Parametr Index

SS Server služeb – Service Server

SSL Secure Sockets Layer

TDMA Vícenásobný přístup s časovým dělením – Time Division Multiple Access

TKIP Temporal Key Integrity Protocol

TGS Server lístků – Ticket Granting Server

TGT Lístek žádosti o lístk – Ticket-Granting Ticket

TMSI Temporary Mobile Subscriber Identity

TLS Transparent Layer Security

WEP Soukromí ekvivalentní drátovým sítím – Wired Equivalent Privacy

WPA WiFi Protected Access

SEZNAM PŘÍLOH

A PŘÍLOHA - OBSAH PŘILOŽENÉHO CD.

HTML – Zdrojové kódy www stránek
Flash – Zdrojové soubory Flash animací
LaTeX – Bakalářská práce v LaTeXu
xmarek40.pdf – Práce ve formátu PDF