

Kryptografická ochrana bezdrátových sítí.
Cryptographic protection of wireless networks.

Bc. Zdeněk Rajnoch

Diplomová práce
2013



Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
akademický rok: 2012/2013

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení: **Bc. Zdeněk Rajnoch**
Osobní číslo: **A11498**
Studijní program: **N3902 Inženýrská informatika**
Studijní obor: **Informační technologie**
Forma studia: **kombinovaná**

Téma práce: **Kryptografická ochrana bezdrátových sítí.**

Zásady pro vypracování:

1. Provedte literární rešerši k tématu práce.
2. Analyzujte možnosti řešení zabezpečení dat v bezdrátových sítích.
3. Formou projektu připravte návrh vhodných řešení ve vybraných situacích.
4. Formou simulace zvoleného prostředí ověřte vhodnost vybraných řešení.
5. Provedte diskusi nad řešením projektu.

Rozsah diplomové práce:

Rozsah příloh:

Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam odborné literatury:

1. DOSTÁLEK, L., VOHNOUTOVÁ, M., KNOTEK, M. Velký průvodce infrastrukturou PKI a technologií elektronického podpisu. Computer Press, 2009. ISBN 978-80-251-2619-6.
2. ČANDÍK, Marek. Základy informační bezpečnosti. vyd. Zlín : Univerzita Tomáše Bati, 2004. 107 s. ISBN 8073182181.
3. BITTO, O. Šifrování a biometrika. BEN, 2005. 168 s. ISBN 80-86686-48-5.
4. KLÍMA, Vlastimil; ROSA, Tomáš. Kryptologie pro praxi – DSA, ECDSA. Dostupné z WWW: [http://crypto-world.info/klima/2004/st_2004_04_21_21.pdf].
5. JAŠEK, R. Ochrana znalostí a dat v podnikových informačních systémech. . UTB, 2002. 250 s. ISBN 80-7318-095-2.

Vedoucí diplomové práce:

doc. Mgr. Roman Jašek, Ph.D.

Ústav informatiky a umělé inteligence

Datum zadání diplomové práce:

22. února 2013

Termín odevzdání diplomové práce:

22. května 2013

Ve Zlíně dne 22. února 2013

prof. Ing. Vladimír Vašek, CSc.
děkan



doc. Mgr. Roman Jašek, Ph.D.
ředitel ústavu

ABSTRAKT

Práce se zabývá zabezpečením bezdrátových sítí z hlediska utajení přenášených dat. Popisuje podstatu a historii kryptografie, principy a metody současných kryptografických metod. Ve druhé části jsou popsány jednotlivé typy bezdrátových sítí a způsoby jejich zabezpečení autentizace a přenosu dat. Třetí část se věnuje jednotlivým kryptografickým algoritmům používaným v bezdrátových sítích a zkoumá jejich nedostatky. V praktické části je řešeno nedostatečné zabezpečení přenosu dat v sítích GSM. Formou programu pro operační systém android je zde nastíněno řešení kryptografického zabezpečení přenosu SMS v síti GSM na aplikační vrstvě.

Klíčová slova:

Bezdrátové sítě, kryptografie, symetrická kryptografie, asymetrická kryptografie, kryptologie, Bluetooth, ZigBee, Wi-fi, GSM, 3G, UMTS, RSA, AES, RC4, A0, E5, Java, Android, DES, TripleDES, Blowfish,

ABSTRACT

The work deals with the security of wireless networks in terms of confidentiality of transmitted data. Describes the nature and history of cryptography, principles and methods of current cryptographic methods. The second part describes the different types of wireless networks and their methods of secure authentication and data transfer. The third part focuses on cryptographic algorithms used in wireless networks and examines their shortcomings. In the practical part is the lack of security of data transmission in GSM networks. The form of the program for android operating system is a cryptographic security solution outlined the SMS in GSM network to the application layer.

Keywords:

Wireless networks, cryptography, asymmetric cryptography, symmetric cryptography, cryptology, Bluetooth, ZigBee, Wi-fi, GSM, 3G, UMTS, RSA, AES, RC4, A0, E5, Java, Android, DES, TripleDES, Blowfish,

Poděkování, motto a čestné prohlášení.

Chtěl bych poděkovat vedoucímu mé diplomové práce panu doc. Mgr. Romanu Jaškovi, Ph.D., za připomínky a poskytnutí cenných informací a zdrojů k této diplomové práci.

Prohlašuji, že jsem na diplomové práci pracoval samostatně a použitou literaturu jsem citoval.

V Olomouci, dne: 5.3.2013.

Podpis

Obsah

ÚVOD	8
I TEORETICKÁ ČÁST	9
1 HISTORIE A PODSTATA KRYPTOGRAFIE	10
1.1 HISTORICKÉ ŠIFRY	10
1.1.1 POČÁTKY KRYPTOLOGIE.....	10
1.1.2 STŘEDOVĚK.....	12
1.1.3 ENIGMA.....	13
1.1.4 KÓD NAVAHŮ	13
1.1.5 SYSTÉM TTS	14
1.1.6 PURPUROVÝ KÓD.....	14
1.1.7 STEGANOGRAFIE	14
1.2 MODERNÍ ŠIFROVACÍ METODY	14
1.2.1 CÍLE KRYPTOGRAFIE	14
1.2.2 SYMETRICKÁ KRYPTOGRAFIE	16
1.2.3 ASYMETRICKÁ KRYPTOGRAFIE.....	18
1.2.4 POSTUPY ŠIFROVÁNÍ DAT	20
1.3 ELEKTRONICKÝ PODPIS	21
2 BEZDRÁTOVÉ SÍTĚ	23
2.1 IRDA	23
2.1.1 SPECIFIKACE	23
2.2 BLUETOOTH	23
2.2.1 SPECIFIKACE	23
2.2.2 PROTOKOLY BLUETOOTH.....	25
2.2.3 ZABEZPEČENÍ PŘENOSU DAT BLUETOOTH	27
2.3 ZIGBEE	30
2.3.1 SPECIFIKACE	30
2.3.2 ZABEZPEČENÍ PŘENOSU DAT.....	31
2.4 WI-FI	31
2.4.1 SPECIFIKACE	31
2.4.2 MOŽNOSTI ZABEZPEČENÍ WI-FI.....	32
2.5 SÍŤ GPS	36
2.6 SÍŤ GSM	36
2.6.1 SPECIFIKACE	36
2.6.2 NMT ,AMPS, TACS, C450 - SÍŤ 1. GENERACE	36
2.6.3 GSM SÍŤ 2. GENERACE	37
2.6.4 SIM KARTA(SUBSCRIBER IDENTITY MODULE)	39
2.6.5 OVĚŘENÍ TOTOŽNOSTI V SÍTI GSM.....	39
2.6.6 ŠIFROVÁNÍ PŘENÁŠENÝCH DAT V SÍTI GSM	42
2.6.7 PŘENOS DAT V SÍTI GSM.....	43
2.7 SÍŤE 3G – UMTS , HSDPA, HSUPA, HSPA+, LTE	45

3	KRYPTOGRAFICKÉ ALGORITMY V BEZDRÁTOVÝCH SÍTÍCH.....	48
3.1	RIJNDAEL.....	48
3.2	RC4.....	50
3.3	E0.....	50
3.4	A5.....	51
3.5	COMP128.....	52
3.6	CRC 32.....	53
3.7	DIFFIE-HELLMANN.....	54
4	NEDOSTATKY ZABEZPEČENÍ SÍTĚ GSM.....	55
4.1	ŠIFROVANÉ TELEFONY.....	55
II	PRAKTICKÁ ČÁST.....	57
5	APLIKACE PRO ŠIFROVÁNÍ SMS NA PLATFORMĚ ANDROID.....	58
5.1	SPECIFIKACE.....	58
5.2	RSA KRYPTOGRAFIE.....	58
5.3	UŽIVATELSKÉ ROZHŘANÍ PROGRAMU.....	59
5.4	TECHNICKÁ DOKUMENTACE.....	61
5.4.1	POPIS TŘÍD.....	61
	ZÁVĚR.....	63
	CONCLUSION.....	64
	SEZNAM POUŽITÉ LITERATURY.....	65
	SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK.....	67
	SEZNAM OBRÁZKŮ.....	69
	SEZNAM TABULEK.....	70
	PŘÍLOHA A:ZDROJOVÝ KÓD RSA.....	71
	PŘÍLOHA B:ZDROJOVÝ KÓD SMS_ENCRYPTOR.....	74

ÚVOD

S rostoucím množstvím uživatelů v počítačových sítích roste i význam zabezpečení dat proti zneužití cizí osobou. V posledních letech roste zejména význam bezdrátových sítí, ať už jsou to sítě standartu IEEE 802.11., nebo IEEE 802.16, případně globální systém pro mobilní komunikaci GSM. Tyto počítačové sítě nabývají na významu zejména díky zvýšené mobilitě účastníků, ale zároveň s sebou nesou větší rizika zneužití dat. Útočníci již nemusí mít přímý přístup k hardware počítačové sítě, data jsou přenášena například pomocí elektromagnetického vlnění, které je možné snadno zachytit pomocí antény. Tato anténa může být vzdálená až několik kilometrů od přístupového bodu. Vzhledem k tomu, že nelze zabránit fyzickému přístupu třetích osob k přenášeným informacím, roste v bezdrátových sítích význam kryptologie tak, aby tato data byla pro nepovolané osoby nečitelná. S rozvojem bezdrátových sítí roste počet bezpečnostních rizik spojených s bezdrátovými protokoly, šifrovacími metodami a nerozvážeností samotných uživatelů. Bezdrátové sítě jsou využívány v mnoha odvětvích, a v případě špatného zabezpečení těchto sítí tak může docházet k růstu počítačové kriminality. Ačkoliv se kryptografické metody během několika posledních desítek let velmi změnily, od poloviny 20. stol. jsou používány složité matematické metody, cíle kryptografie zůstávají stále stejné. Historii kryptografických metod popisuje první kapitola. Cílem je podat ucelený přehled kryptografických metod. Druhá kapitola se bude zabývat typy bezdrátových sítí, budou popsány principy jejich kryptografického zabezpečení při autentizaci a přenosu dat. Třetí část se bude zabývat algoritmy používanými v bezdrátových sítích. Pokusím se odhalit nedostatky v zabezpečení těchto sítí, zejména v nejpoužívanější bezdrátové síti GSM. Tato síť je již dnes poměrně zastaralá. Přenos dat v této síti není bezpečný. Řešení tohoto problému představují například šifrované telefony, které však jsou poměrně drahé. Proto se v praktické části pokusím vytvořit program pro operační systém Android v jazyce Java, který bude umět šifrovat přeposílané sms mezi dvěma účastníky tak, aby třetí strana nedokázala rozluštit jejich obsah. Operační systém Android jsem vybral proto, že je to moderní operační systém, který se v současné době rychle rozvíjí a má největší procentuální podíl na trhu, co se týká mobilních telefonů.

I. TEORETICKÁ ČÁST

1 HISTORIE A PODSTATA KRYPTOGRAFIE

Kryptografie je věda o utajování smyslu zpráv převodem do podoby, která je čitelná jen pomocí znalosti jak dostat zpět původní zprávu. Kryptografie se vyvíjela po staletí, její největší význam byl zejména v utajování vojenských informací. Historie kryptografie se dá rozdělit do dvou období, první období trvalo do poloviny 20. století. Do té doby bylo šifrování velmi jednoduché, tyto šifry by v dnešní době vlivem výpočetní techniky byly neúčinné. Druhé období je charakteristické rozvojem výpočetní techniky a moderních matematických metod. K šifrování je v dnešní době používáno kryptografických algoritmů, které převádí čitelnou zprávu na nečitelnou. Za použití klíče a znalosti šifrovacího algoritmu je potom možné dostat ze zakódované zprávy zpět zprávu původní. V následujících kapitolách jsou popsány kryptografické algoritmy, které byly používány v různých obdobích historie, až po nejmodernější metody založené na složitých matematických principech využívaných k přenosu dat pomocí výpočetní techniky a počítačových sítí.

1.1 Historické šifry

1.1.1 Počátky kryptologie

Počátky aplikace kryptologie, tak jak ji známe v současné době, se datují až do roku 1900 př. n. l., kdy staří Egypťané používali pro zápis citlivých dat atypické hieroglyfy. Pomocí takto zapsaných značek měli zajištěno, že k důvěrným informacím bude mít přístup pouze předem určená skupina lidí. Nutno však poznamenat, že i použití běžných hieroglyfů se dá považovat za jistou formu šifrování, jelikož mnoho lidí neumělo hieroglyfy číst, natož je pak třeba i psát [1].

O rozvoj v oblasti šifrování se velkou měrou zasloužili staří Řekové. Již kolem roku 350 př. n. l. navrhl vojevůdce Aeneus Tacticus okolo dvaceti šifrovacích klíčů rozdělených do dvou skupin – transpoziční a substituční. Řecký historik Plutarchos zdokumentoval vznik prvního transpozičního šifrovacího systému (písmena otevřeného textu jsou přeskupována podle předem přesně určených pravidel), zvaného SKYTALA, který používali spartánští vojevůdci kolem roku 500 př. n. l. Princip byl jednoduchý. Na dřevěnou hůl o přesném průměru se navinula tenká kožená nebo pergamenová páska a napsala se na ni tajná zpráva. Poté byla páska opět sejmuta z hole. Pouze osoba, která vlastnila dřevěnou hůl o stejném průměru, mohla přečíst šifrovanou zprávu. Při pokusu o přečtení zprávy na holi o jiném průměru útočník získal pouze nesmyslnou změť písmen [1].

Údajně byl praktikován také tento zajímavý způsob utajení informace: vybranému otrokovi byla ostříhána hlava dohola a poté zpráva určená k utajenému přenosu vytetována na lebku. „Odesílatel“ utajované zprávy poté počkal, než otrokovi opět dorostly vlasy a poslal jej k „příjemci“. Ten si po oholení otrokovy hlavy přečetl zprávu. Jedinou nevýhodou byla nemožnost opakovaného použití otroka... [1].

Velkým objevem byla tzv. Polybiova šifrovací mřížka. Její princip byl stejně jako u většiny antických šifer, velice jednoduchý. Abeceda byla vepsána do obdélníkové mřížky, kde bylo každé písmeno reprezentováno dvojicí čísel, na jejichž průsečíku řádku a sloupce se dané písmeno nacházelo [1].

Slavnou se stala Caesarova šifra. Ta pochází z roku 50 př. n. l. Jedná se o klasický substituční systém (znaky otevřeného textu jsou nahrazovány jinými znaky, dle předem dohodnutého systému). Caesar tuto šifru používal i při dopisování s egyptskou královnou Kleopatrou. Princip Caesarovy šifry byl jednoduchý: šifrování probíhalo tak, že se každý znak nahradil znakem, který je v abecedě o 3 pozice před ním. Algoritmus pro dešifrování byl analogický: každý znak se nahradil znakem, který je o tři pozice za ním. Zároveň se jedná o příklad tzv. monoalfabetické šifry. Později byla Caesarova šifra zdokonalena proměnlivou hodnotou posunutí v abecedě [1].

Po tomto kryptologicky relativně plodném období následovalo několik století útumu. Kolem roku 750 n. l. byla v Byzantské říši napsána kniha pojednávající o základních šifrovacích metodách a postupech, která byla určena pro tamějšího vládce. Roku 855 bylo ve stejné oblasti publikováno několik šifrovacích klíčů, které však sloužili výhradně pro účely magie. Arabové zároveň položili základy kryptoanalýzy, kdy zkoumali frekvenci výskytu jednotlivých hlásek ve slovech. Princip byl jednoduchý. Kryptoanalytik si vzal libovolný relativně dlouhý otevřený text a spočítal výskyt jednotlivých hlásek. Na základě takto získané statistiky poté spočítal a nahradil znaky v nějaké jednoduché substituční šifře. Úspěšnost tohoto útoku na šifrovaný text byla při použití jednoduchého šifrovacího algoritmu relativně vysoká [1].

V Evropě se kryptografie začala rozvíjet o něco později. V roce 1379 byla vydána sbírka obsahující několik známých šifrovacích algoritmů [1].

1.1.2 Středověk

Za průkopníka modernější kryptografie je považován benediktinský opat ze Spanheimu Johannes Trithem (1462-1518). Ten v roce 1518 napsal knihu jménem Stenografie. Principem stenografické šifry je nahrazení každého písmene slovem v předem určené tabulce. Stejněho roku vydal první tištěnou knihu zabývající se popisem některých známých šifrovacích algoritmů (s důrazem kladeným na substituční systém). Její název byl „Polygraphiae libri sex“ a byla opakovaně vydávána v letech 1550, 1564, 1571 a 1600. V letech 1561 a 1564 byla vydána její francouzská verze. Tehdejší panovnické rody se však obávali, že by mohl vyzradit příliš mnoho tajemství, tak jej pro jistotu označili za čarodějníka spojeného s ďáblem. Roku 1526 Jacopo Silvestri vydal knihu s názvem „Opus novum...principibus maxime vtilissimum pro cipharis“, v níž popsal šest šifrovacích metod (včetně velice oblíbené Caesarovy šifry). Kniha byla psána pro praktické použití a mohla tak oslovit širokou čtenářskou obec [1].

V roce 1550 byla vydána kniha s názvem De subtilitate libri XIII. Jejím autorem byl Girolamo Cardano a dočkala se až nečekaného úspěchu-celkem osm vydání [1].

Další významnou osobností v oblasti kryptologie se stal Ioan Battista Porta. Ten roku 1563 vydal čtyřdílný spis s názvem De fvtivis litararvm notis, vvlgo de ziferis Libri III. V nich popsal rozdělení šifer na transpoziční, substituční a substituční symbolové (použití nestandardní abecedy). Zároveň také navrhl použití různých „nadbytečných“ znaků, které měli ztížit práci kryptoanalytikům při případném útoku na šifrovaný text [1].

Italský renesanční učenec Jeroným Cardan vynalezl nový způsob šifrování, který je v současné době znám pod názvem Cardanova mřížka. Šifrovací mřížkou byla v podstatě destička, ve které byly na určitých místech vyřezány otvory. Odesílatel přiložil tuto mřížku na papír a do průřezů vepsal zprávu k odeslání. Poté mřížku sejmul a text libovolně doplnil. Pouze příjemce, který měl identickou mřížku, byl schopen jejím přiložením šifrovaný text „převést“ do otevřené podoby [1].

Transpoziční šifru používal také francouzský kardinál Richelieu. Ten přeskupoval písmena v textu pomocí klíčového slova, které definovalo změnu pozice písmen [1].

V roce 1586 vydal Blaise de Vigenère svou knihu Traicté des chiffres, která měla 600 stran. Popsal v ní mnoho šifrovacích algoritmů včetně praktických ukázek [1].

Nejznámějším anglickým autorem v oblasti kryptologie se stal Francis Bacon, který vytvořil tzv. bigramovou substituci (bigram = dvojice po sobě jdoucích znaků), dnes zná-

mou jako pěti bitové binární kódování. Jeho nejznámějším dílem je kniha z roku 1623 s názvem Proficience and Advancement of Learning Divine and Humane [1].

Jedním z největších vynálezů byla Morseova abeceda. Ve své podstatě se nejedná o klasickou šifru, ale spolu s vynálezem telegrafu činila velký průlom v oblasti kryptografie. Jednalo se o první šifrovací systém, ve kterém odesílatel a příjemce mohli rychle komunikovat na dálku bez potřeby třetí strany k přenosu šifrované informace. Tím odpadli starosti s „bezpečností přenosu“. Do této doby bylo vždy potřeba zajistit bezpečný přenos šifrované informace a šifrovacího klíče. Ne zřídkakdy se tak stávalo, že posel vyslaný s takovýmto materiálem byl přepaden a fyzicky donucen k prozrazení tajných informací, či mu byl ukradnut šifrovací klíč [1].

1.1.3 Enigma

Enigmou si nechal patentovat 18. 2. 1918 německý inženýr Arthur Scherbius a v dubnu téhož roku ji nabídl německému námořnictvu. V letech 1926 a 1928 německé námořnictvo používalo upravenou verzi stroje Enigma. V roce 1928 se jeden exemplář Enigmy dostal díky problémům při transportu do rukou polským matematikům ve Varšavě. Ti přišli na to, že klávesnice je spojována s kódovacím zařízením v abecedním pořadí. Na základě tohoto zjištění sestrojili dekodovací zařízení La Bombe [1].

Šifru již začátkem 30. let prolomili polští kryptoanalytici. Po obsazení Polska Německem navázali na jejich práci kryptoanalytici britští, kteří po celou dobu 2. světové války úspěšně četli tajné depeše nepřítele bez pomoci enigmy. Pro mylný názor, že jím vytvořené šifry jsou nerozluštitelné, používaly tento stroj a jeho menší modifikace i některé vlády ještě v 50. letech 20. století, včetně Sovětského svazu.

1.1.4 Kód Navahů

Američané během války v Tichomoří sáhli po originálním postupu šifrování. Ke komunikaci využili jazyk indiánského kmene Navahů. Přenos zprávy probíhal jednoduchým způsobem, každá jednotka měla radistu – Navaha. Ten převzal zprávu v angličtině, poté ji přeložil do navažštiny, v navažštině ji posílal druhému radistovi, který zprávu zpět přeložil do angličtiny. Poměrně jednoduchý způsob, ale velmi efektivní. Někteří odborníci později způsob komunikace kritizovali, ale během druhé světové války byl úspěšný [2].

1.1.5 Systém TTS

Sehrál rozhodující úlohu při průniku německých luštitelů do československé londýnské zpravodajské sítě. Fatální chybou zpravodajců při jeho rozluštění bylo předávání šifrovaného klíče stejným kanálem jako šifrované zprávy. Byl používán během druhé světové války od roku 1939 do roku 1941 [2].

1.1.6 Purpurový kód

Představuje jeden z nejslavnějších šifrovacích systémů. Japonci začali tento kód používat od roku 1937. Jednalo se o komplexní systém, jeho rozluštění vyžadovalo vysoké intelektuální předpoklady. Tento kód rozluštili američané v roce 1940 [3].

1.1.7 Steganografie

Věda o utajení komunikace prostřednictvím ukrytí zprávy. Do oblasti steganografie patří například neviditelné inkousty, mikrotečky případně schovávání zpráv do souborů s obrázky nebo zvukových souborů v podobě náhodného šumu. Steganografie je založena na tom, že si pozorovatel vůbec neuvědomuje probíhající komunikaci. V případě že je prozrazena komunikace, dojde i k prozrazení přenášené zprávy.

1.2 Moderní šifrovací metody

1.2.1 Cíle kryptografie

Ačkoliv se kryptografické metody během několika posledních desítek let velmi změnily, od poloviny 20. stol. jsou používány složité matematické metody, cíle kryptografie zůstávají stále stejné.

Cíle kryptografie:

- důvěrnost – udržení obsahu zprávy v tajnosti
- celistvost dat – zamezení změny obsahu dat
- autentizace dat – ověření, že ten s kým komunikujeme je skutečně ten, za koho se vydává
- autorizace – potvrzení původu dat, ověření autora
- nepopíratelnost – jedná se o jistotu, že autor dat nemůže své autorství popřít

Moderní kryptografické metody na rozdíl od těch starověkých se řídí „Kerckhoffsovým principem“, což znamená, že bezpečnost šifrovacího systému nesmí záviset na utajení algoritmu, ale pouze na utajení šifrovacího klíče [4].

Prakticky každý kryptografický algoritmus je založený na nějakém matematickém problému, který má výpočetně obtížné řešení. Asymptotickou výpočetní složitost algoritmů dělíme na časovou a prostorovou. Při pokusech o rozlomení šifry nás bude zajímat časová složitost. Asymptotická složitost je způsob, jakým klasifikujeme počítačové algoritmy. Je udávána pomocí funkce, která řádově určuje množství operací potřebných k provedení výpočtu vzhledem k velikosti vstupní hodnoty do algoritmu. Přehled těchto složitostí je v tabulce 1.

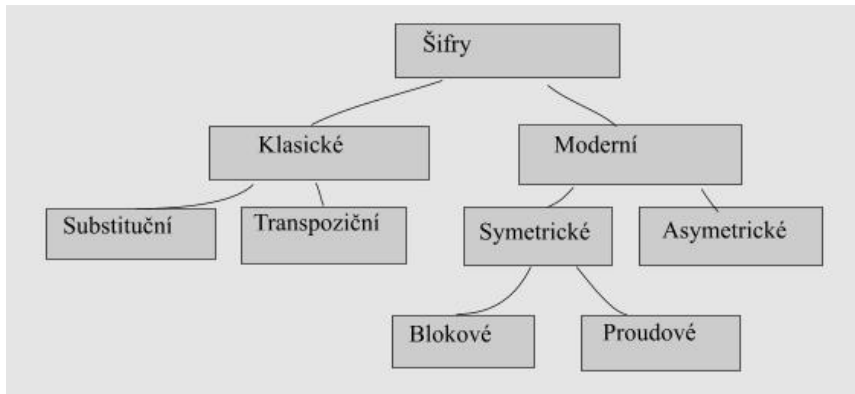
Velikost vstupu	Třída složitosti							
	Konstantní	Logaritmická	Lineární	Lineárně logaritmická	Kvadratická	Polynomiální	Exponenciální	Faktoriálová
1	1	1	1	1	1	1	2	1!
10	1	1	10^1	$2 \cdot 10^1$	10^2	10^3	2^10	10!
100	1	2	10^2	$2 \cdot 10^2$	10^4	10^6	2^100	100!
1000	1	3	10^3	$3 \cdot 10^3$	10^6	10^9	2^1000	1000!
1 000 000	1	6	10^6	$6 \cdot 10^6$	10^{12}	10^{18}	$2^{1000000}$	1000000!

Tabulka 1: Třídy složitostí

Tabulka 1 ukazuje, jak se u algoritmů s danou složitostí mění doba výpočtu vzhledem ke vstupní hodnotě algoritmu. Algoritmy použitelné v praxi mají maximálně polynomiální složitost. U exponenciální a faktoriálové složitosti bychom se výsledku v reálném čase pro vyšší hodnoty nedočkali. Právě této skutečnosti využívají kryptografické algoritmy, kde zašifrování pomocí klíče je velice rychlé, ale rozlomení šifry bez pomoci klíče má exponenciální složitost a trvá tedy velice dlouhou dobu. U systémů pro šifrování dat je velice důležitá implementace daného algoritmu. Většina útoků není vedena přímo na kryptografický algoritmus, ale spíše na jeho implementaci. U systémů na šifrování dat sledujeme několik hledisek:

- u šifrování by nemělo narůstat množství dat
- implementace šifrovacích algoritmů by měla být co nejjednodušší
- algoritmus by měl být co nejrychlejší
- neměl by mít omezení na vstupní data
- algoritmus by měl pracovat bezchybně

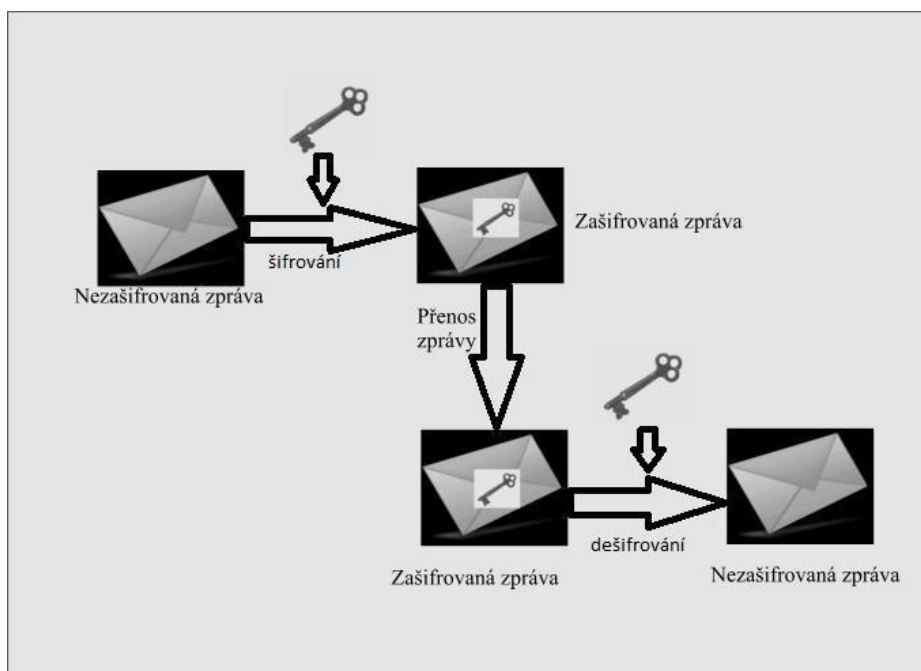
U algoritmů lze sledovat různé vlastnosti, které se týkají předcházejících bodů. Na obrázku 1 je znázorněno rozdělení šifer.



Obrázek 1: Rozdělení šifer

1.2.2 Symetrická kryptografie

Tyto šifrovací algoritmy používají jeden klíč. Podstatnou výhodou těchto algoritmů je jejich velká rychlost. Nevýhodou těchto algoritmů je, že obě strany, které přeposílají zprávy se musí předem domluvit na tomto klíči. Symetrické šifry se dělí na dva druhy. Proudové šifry zpracovávají otevřený text po jednotlivých bitech. Blokované šifry rozdělí otevřený text na bloky stejné velikosti a doplní vhodným způsobem poslední blok na stejnou velikost. U většiny šifer se používá blok o 64 bitech, AES používá 128 bitů. Princip symetrické kryptografie je na obrázku 2. Tyto šifry a jejich využití si krátce popíšeme v následujících kapitolách.



Obrázek 2 : Princip symetrického šifrování

1.2.2.1 Data Encryption Standard – DES

Standart, který byl původně publikován v NBS-National Bureau of Standards, což je pobočka ministerstva obchodu v USA. DES byl původně navržen jako standart IBM. Vycházel z šifry LUCIFER, kterou vyvinul Horst Feistel. K rozvoji DES přispěla americká vláda a organizace NASA. DES byl přijat jako standart a publikován v roce 1977 jako FIPS – Federal Standard Information Processing.

1.2.2.2 Triple DES

Původní DES má délku klíče 56 bitů, což se postupem času stalo nedostatečným a klasický DES tak byl ohrožen útoky hrubou silou. Triple DES je nejjednodušším způsobem, jak odolnost DESu zvýšit díky většímu klíči bez nutnosti přejít na zcela nový algoritmus. Triple DES je trojnásobnou aplikací šifry DES. Od jeho používání se dnes již upouští, protože trojnásobné použití šifry způsobuje, že je tento algoritmus pomalý.

1.2.2.3 AES – The Advanced Encryption Standard

S neustálým pokrokem výpočetní techniky již není DES vhodné pro elektronické transakce. AES se tak stal novým standartem pro elektronické obchodování v roce 1997. Pro tento standart byl vybrán algoritmus Rijndael. Tento algoritmus navrhl Joan Daemen a Vincent Rijmen. Tento algoritmus šifruje bloky o velikosti 128 bitů s klíči o velikosti 128, 192 nebo 256 bitů [5].

1.2.2.4 Blowfish

Je symetrická bloková šifra, používá bloky dat o délce 64 bitů a klíč proměnné délky od 32 do 448 bitů. Je považována za jednu z nejbezpečnějších šifer. Blowfish je používána jako standartní šifra v operačním systému FreeBSD. Tato šifrovací metoda nepodléhá žádnému patentu a může se volně používat.

1.2.2.5 RC2, RC5

Šifry navrhl v roce 1987 a 1994 Ron Rivest. RC2 je 64bitová bloková šifra s proměnnou velikostí klíče. RC5 má proměnnou blokovou velikost 32, 64 nebo 128 bitů.

1.2.2.6 IDEA

Jeden z nejlepších algoritmů. Používá bloky o 64 bitech a 128 bitový klíč. Různé skupiny kryptologů prováděly kryptoanalýzu šifrovacího algoritmu IDEA, ale zatím žádná nepřišla

se zveřejněním nějakých slabín algoritmu. Algoritmus IDEA je při šifrování přibližně dvakrát rychlejší než algoritmus DES (a současně nabízí podstatně vyšší úroveň bezpečnosti). IDEA je patentována v USA a ve většině evropských zemí (patent neplatí ve Finsku). Majitelem patentu je firma Ascom-Tech. Nekomerční použití algoritmu je bezplatné [6].

1.2.2.7 GOST 28147-89

Tento kryptografický algoritmus je určen buď pro hardwarové, nebo softwarové implementace. Tento algoritmus pracuje s 64 bitovými bloky dat. Používá 256 bitový klíč. Tento algoritmus je alternativou pro AES-256 a Triple DES. Je však mnohem jednodušší na implementaci. GOST je implementován v standardní kryptografické knihovně, jako jsou OpenSSL a Crypto ++ [7].

1.2.2.8 Twofish

Twofish je symetrická bloková šifra s 128bitovou délkou bloku a až 256bitovou délkou klíče, vyvinutá Brucem Schneierem. Jde o nepatentovanou otevřenou šifru pro volné použití. Šifra Twofish byla jedním z pěti finalistů soutěže standardu AES.

1.2.2.9 Skipjack

Šifra vyvinutá U. S. National Security Agency (NSA). Byla původně určena pro použití v Clipper čipu. Používá 80bitový klíč k šifrování a dešifrování 64bitových datových bloků.

1.2.2.10 Fish

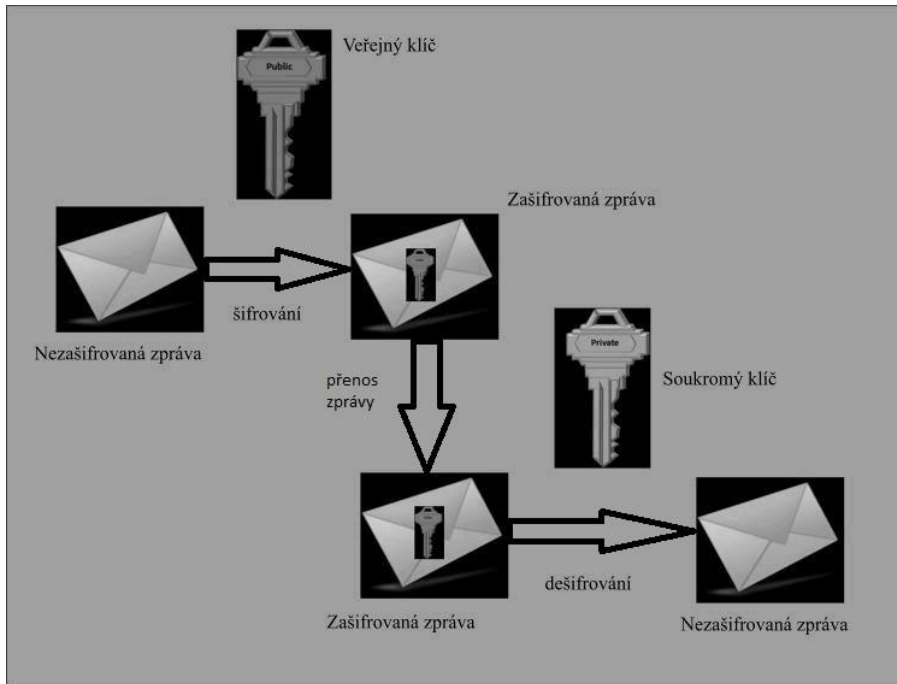
Patří mezi symetrické proudové šifry. Je založena na Fibonacciho generátoru pseudonáhodných čísel. Algoritmus byl publikován v roce 1993, ale nebyl nikdy moc používán.

1.2.2.11 RC4

Symetrická proudová šifra, kterou vymyslel Ron Rivest. Šifra má velice rychlý a jednoduchý algoritmus.

1.2.3 Asymetrická kryptografie

Asymetrická kryptografie vznikla v roce 1975. Skupina kryptografických metod, u kterých se pro šifrování a dešifrování dat používají dva odlišné klíče. Veřejný klíč se používá k šifrování dat. Soukromý klíč vlastní jen příjemce zprávy, je používán k dešifrování dat. Princip asymetrické kryptografie je na obrázku 3.



Obrázek 3: Princip asymetrické kryptografie

Asymetrická kryptografie je mnohem náročnější na výpočetní výkon než kryptografie symetrická. Je založena na principu jednosměrných funkcí. Využívá toho, že dosud nebyla objevena metoda, jak rozložit velká čísla na prvočísla.

1.2.3.1 RSA

Bezpečnost RSA je postavena na předpokladu, že rozložit velké číslo na součin prvočísel je výpočetně velmi náročné. Z čísla $n = p * q$ je tedy v rozumném čase nemožné zjistit činitele p a q , neboť není znám žádný algoritmus faktorizace, který by pracoval v polynomiálním čase vůči velikosti binárního zápisu čísla n . Naproti tomu násobení dvou velkých čísel je velice snadné. RSA je jedna z nejpoužívanějších metod. RSA vzniklo v roce 1978.

1.2.3.2 ElGamal

Jeden z algoritmů asymetrické kryptografie, má ovšem nevýhodu, že šifrovaná data jsou dvakrát delší než data nešifrovaná. To je možná důvodem, proč není používán jako algoritmus RSA, který tímto nedostatkem netrpí. Opírá se o problém výpočtu diskretního logaritmu.

1.2.3.3 Kryptografie eliptických křivek

Je metoda šifrování veřejných klíčů založená na algebraických strukturách eliptických křivek nad konečnými poli. Použití eliptických křivek v kryptografii navrhli nezávisle na sobě Neal Koblitz a Victor S. Miller v roce 1985.

1.2.3.4 Diffie-Hellman

Tento algoritmus je bez kombinace s jinými metodami vhodný pouze tam, kde útočník nemůže aktivně zasahovat do komunikace. Klíč je zkonstruován všemi účastníky komunikace a nikdy není poslán v otevřené formě. Tento algoritmus se opírá o složitost výpočtu diskrétního logaritmu.

1.2.3.5 DSA – Digital Signature Algorithm

Standart americké vlády pro digitální podpis. Algoritmus je založen na problému výpočtu diskrétního logaritmu. DSA je patentováno.

1.2.4 Postupy šifrování dat

Při přenosu dat bezdrátovou sítí hrozí dvě základní nebezpečí.

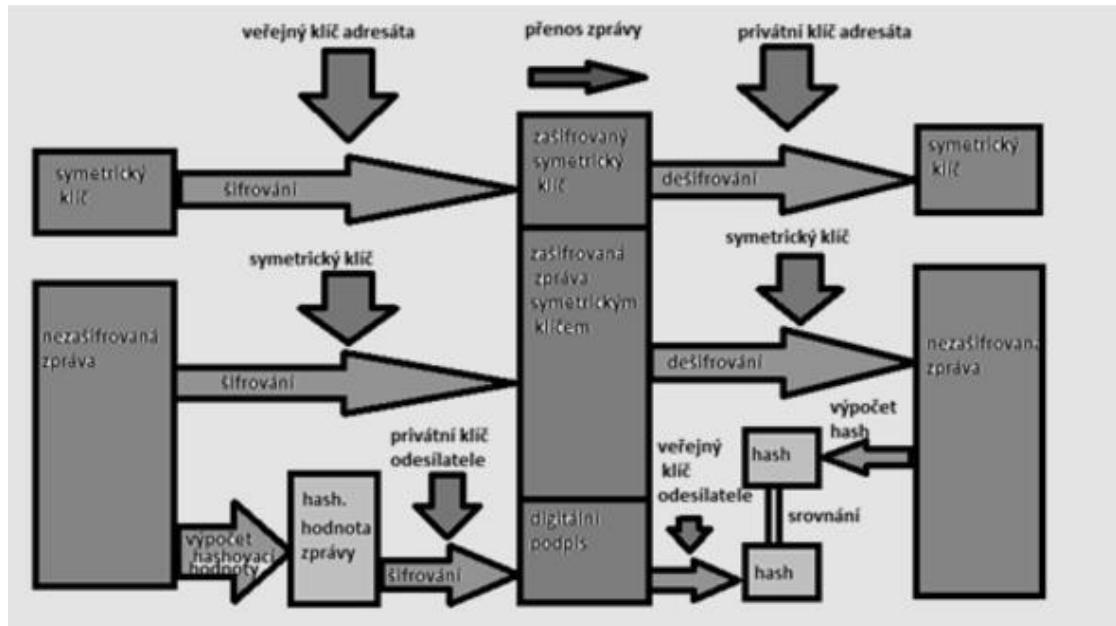
- nebezpečí modifikace dat
- nebezpečí zneužití dat

1.2.4.1 Nebezpečí modifikace dat

Nejvhodnější ochranou proti modifikaci dat je digitální otisk. Digitální otisk je kontrolní součet přenášeného souboru. Tento otisk se odešle zároveň s přenášeným souborem, a pokud odpovídá kontrolní součet souboru před odesláním a po odeslání je velká pravděpodobnost že tyto soubory jsou shodné.

1.2.4.2 Nebezpečí zneužití dat

K šifrování přenášených dat se používají symetrické šifry, kde uživatel zná heslo, které nesmí znát nikdo jiný. Algoritmy symetrické kryptologie jsou velice rychlé, ale problém vzniká s doručením klíče symetrické šifry příjemci. Asymetrická kryptologie se proto využívá k zašifrování klíče symetrické kryptologie a k přeposlání takto zašifrovaného klíče příjemci. Takto doručení klíč symetrické šifry se použije k samotnému šifrování přenášených dat. Princip šifrování dat je znázorněn na obr. 4.



Obrázek 4: Kombinace symetrické a asymetrické kryptografie

1.2.4.3 Hašovací funkce

Používají se v symetrické i asymetrické kryptografii. Jedná se o funkce, které z libovolného souboru dat vypočítají tzv. „hash“, což je sekvence bajtů, ze kterého se dá určit, jestli při přenosu souboru nedošlo k jeho změně. Kromě toho se používají například k vyhledávání dat v databázích. Asymptotická složitost takového vyhledávacího algoritmu je konstantní. Hašovací funkce mají tyto vlastnosti:

- jakékoliv množství vstupních dat poskytuje stejně dlouhý výstup
- malou změnou vstupních dat dosáhneme velkou změnu na výstupu
- z hashe je prakticky nemožné rekonstruovat původní text zprávy, funkce jsou jednosměrné
- v praxi je vysoce nepravděpodobné, že dvěma různým zprávám odpovídá stejný hash
- z definice funkce plyne možnost tzv. kolizí, což znamená, že dva různé soubory mohou mít stejný otisk, avšak v praxi je to málo pravděpodobné

1.3 Elektronický podpis

Je označení specifických dat, která v informatice nahrazují ruční podpis. Stejně jako asymetrická kryptografie využívá dvou klíčů, s tím rozdílem, že ze souboru dat se nejdříve vypočítá hash, který se následně zašifruje soukromým klíčem a připojí se k těmto datům. Pro elektronický podpis se využívají algoritmy RSA, DSA, MD5 a SHA.

Elektronický podpis umožňuje ověřit několik skutečností:

- autenticita
- integrita
- nepopiratelnost
- časové ukotvení

Důvěryhodnost elektronického podpisu je však ještě nutné zajistit pomocí „certifikační autority“, což je důvěryhodná třetí strana, která certifikáty vydává. V případě že je k podpisu využita symetrická kryptografie pak mluvíme o tzv. „Arbitrovaném protokolu“.

2 BEZDRÁTOVÉ SÍTĚ

Bezdrátová síť je typ sítě, kde je přenos dat uskutečňován jiným způsobem než pomocí elektrických vodičů. Tyto počítačové sítě lze rozdělit na dva hlavní typy. Prvním typem jsou sítě s malým dosahem, což jsou technologie určené pro budování lokálních sítí sloužících pro připojení několika zařízení. Jsou to například technologie Bluetooth, ZigBee nebo IrDa. Druhou skupinou jsou sítě s větším dosahem určené primárně pro připojení lokálních sítí k internetu, nebo propojení budov. Tento přenos dat bývá uskutečněn pomocí elektromagnetických vln. Dalšími příklady bezdrátových sítí může být systém GPS nebo nejpoužívanější bezdrátová technologie GSM.

2.1 IrDA

2.1.1 Specifikace

Tato technologie je specifikována IrDA konzorciem, která specifikuje přenos dat pomocí infračerveného záření. Tento standart vznikl z potřeby propojit dvě zařízení pro přenos dat mezi sebou bez propojení elektrických kabelů. Tato technologie pracuje do vzdálenosti přibližně 1m. při chybovosti 10^{-9} , což je dáno souosostí IrDA vysílačů obou zařízení. Přijímačem jsou PIN fotodiody, které pracují v generačním režimu, což znamená, že při dopadu světla na přijímač vytvoří světlo elektrony, které se odvádí do elektrického filtru, který propustí jen ty frekvence které jsou povoleny pro daný typ IrDA modulace. Tato technologie je ovšem ve vysoké míře vytlačována technologií Bluetooth a Wi-Fi. IrDA nepoužívá žádný typ kryptografického zabezpečení dat, což znamená, že veškeré zabezpečení je dáno aplikací, pomocí které jsou data přenášena.

2.2 Bluetooth

2.2.1 Specifikace

Technologie je definovaná standartem IEEE 802.15.1. Bezdrátovou technologii Bluetooth využívá široká škála výrobků, od automobilů a mobilních telefonů až po zdravotnické zařízení a počítače. Pomocí technologie Bluetooth můžete sdílet hudbu, fotky, videa, hlasové a další informace bezdrátově mezi dvěma spárovanými přístroji. Bluetooth technologie využívá rádiové vlny. Největším rozdílem mezi Bluetooth technologií a zařízeními, jako FM rádia a TV je vzdálenost. Rádia a televize jsou určeny k vysílání mnoha lidem na vzdálenost mnoha kilometrů. Technologie Bluetooth odesílá informace přímo ve Vašem osobním

prostoru, který se nazývá „Personal Area Network“ neboli „PAN“ na vzdálenost až 50 metrů. Bluetooth technologie byla vynalezena v roce 1994 inženýry švédské společnosti Ericsson. V roce 1998, se několik společností dohodlo na spolupráci a začaly používat technologii Bluetooth jako způsob, jak propojit své výrobky. Tyto společnosti tvořily „Bluetooth Special Interest Group“ (SIG), organizaci která se snažila dále rozvíjet tuto technologii. To znamená, že tuto technologii nevlastní žádná soukromá společnost, ale že technologii Bluetooth vyvíjí mnoho společností organizace Bluetooth SIG. Tato technologie původně vznikla jako náhrada za propojení domácích zařízení propojovacími kabely. Technologie Bluetooth se nadále vyvíjí a s její pomocí lze vytvořit nové připojení, které nebyly možné pomocí vodičů, jako je připojení mobilního telefonu k autorádiu, nebo např. tisk obrázku přímo z mobilního telefonu [8].

Hlavní rysy technologie Bluetooth je robustnost, nízká spotřeba a nízké náklady. Specifikace Bluetooth definuje jednotnou strukturu pro širokou škálu zařízení pro připojení a komunikaci s ostatními. Zařízení tak pomocí technologie Bluetooth můžou vytvářet sítě tzv. „Piconets“. Piconet je ad hoc počítačová síť zařízení používajících technologii Bluetooth k tomu, aby se jedno hlavní zařízení mohlo spojit s až sedmi vedlejšími zařízeními. Důvodem tohoto omezení je tříbitová MAC adresa. Až 255 dalších vedlejších zařízení může být neaktivních nebo vyčkávajících, až je hlavní zařízení vyzve ke komunikaci. Základní síla bezdrátové technologie Bluetooth, spočívá ve schopnosti současně zpracovávat data a hlasové přenosy. Tato možnost poskytuje uživatelům řadu inovativních řešení, jako je například hands-free sluchátka pro hlasové hovory, tisk a faxování, nebo třeba synchronizace pro počítače a mobilní telefony. Na rozdíl od jiných bezdrátových standardů, specifikace Bluetooth jádro používá k přenosu aplikační vrstvu, která podporuje datové a hlasové aplikace. Technologie Bluetooth pracuje v bezlicenčním průmyslovém, vědeckém a lékařském (ISM) pásmu na 2,4 až 2,485 GHz. Toto pásmo je volně k dispozici ve většině zemí [8].

První verze Bluetooth 1.0 se objevila v roce 1999. Další verze 1.1 potom v roce 2000 ale tyto verze se moc nerozšířily díky vysoké ceně. Až teprve s verzí 1.2 se Bluetooth velice rychle rozšířilo po celém světě. V roce 2004 nastupuje verze 2.0 a v roce 2007 dosud používaná verze 2.1. Technologie EDR (Enhanced Data Rate) je k dispozici od verze 2.0 a používá se až dosud. Úprava 3.0 byla dokončena v dubnu 2009, nějakou chvíli potom trvalo, než se dostala do výroby. Zařízení Bluetooth je velice praktické, mělo však jeden nedostatek, což je přenosová rychlost. Tento nedostatek odstraňuje až verze 3.0, u které je

přenosová rychlost až 24Mbit/s na aplikační vrstvě. Další změnou u verze 3.0 je rozšíření kmitočtového pásma přidáním pásma 5GHz. Díky tomuto rozšíření lze využívat připojení wi-fi pokud je toto rozhraní implementované výrobcem. Ve verzi 3.0 přibylo nativní AES 128bitové šifrování, což velice výrazně pomůže bezpečnosti provozu a zároveň jeho rychlosti. V červenci 2010 přichází verze 4.0, která má stejnou přenosovou rychlost jako předchozí verze, ale ubírá se spíše směrem snižování spotřeby [9].

Jak již bylo zmíněno, Bluetooth využívá ke komunikaci aplikační vrstvu síťového protokolu. Z toho vyplývá, že pro každý typ připojitelného zařízení musí mít Bluetooth definován speciální protokol pomocí kterého s ním bude komunikovat. Tento způsob komplikuje vývoj softwarové podpory Bluetooth (tj. ovladač zařízení), ale i kompatibilitu jednotlivých implementací, jež mohou obsahovat chyby, které způsobí nefunkčnost komunikace. Na druhou stranu však zjednodušují vývoj software, který dané zařízení používá a konfiguraci jednotlivých zařízení, která mají být propojena. Topologie sítě Bluetooth je hvězdicová. Jedno zařízení je typu „Master“, k němu může být připojeno až sedm dalších zařízení „Slave“. Zařízení typu „Slave“ nemůžou komunikovat přímo mezi sebou, ale můžou být měněny ze „Slave“ na „Master“. Každé zařízení může být připojené až do dvou pikosítí. Master zařízení se stará o přidělování komunikačních kanálů.

2.2.2 Protokoly Bluetooth

V systému Bluetooth jsou definovány tyto protokoly:

- Bluetooth Radio (layer)
- Baseband
- LMP
- L2CAP
- RFCOMM
- Service Discovery Protocol (SDP)

2.2.2.1 Bluetooth Radio (layer)

Nejnižší definovaná vrstva Bluetooth specifikace. Definuje požadavky na zařízení Bluetooth transceiveru působící v pásmu 2,4 GHz ISM [10].

2.2.2.2 Baseband

Spravuje fyzické kanály, tento protokol je implementován jako Link Controller, který pracuje s Link Managerem. Provádí rutinu na úrovni spojení a ovládání síly elektrického signálu. Baseband také řídí asynchronní a synchronní vazby, zpracovává pakety, provádí stránkování a vytváří dotazy pro zjišťování dalších zařízení v okolí. [10]

2.2.2.3 LMP – Link Manager Protocol

Funkce LMP se dají rozdělit do několika oblastí:

- správa spojení, což jsou funkce, které ruší, nastavují a spravují spojení mezi zařízeními
- bezpečnostní management – řízení párování, výměna klíčů, autentizace a šifrování
- správa sítě – parametry časování a požadavky na jméno
- konfigurace spojení – kontrola kvality spojení, volba typů paketů, řízení vysílacího výkonu RSSI [10].

2.2.2.4 L2CAP

Definuje spojení pro provozní kanály ACL, tyto kanály jsou analogií portů u protokolů TCP/IP, které tvoří společně s IP adresou socket. U L2CAP je kanál jednoznačně identifikován adresou zařízení a identifikátorem kanálu, který je přidělen potřebě aplikaci která jej používá [11].

2.2.2.5 RFCOMM

Tento protokol poskytuje emulaci sériových portů přes protokol L2CAP. Protokol je založen na ETSI standardu TS 07.10. Používá pouze část TS 07.10 standardu, a některé úpravy protokolu jsou uvedeny ve specifikaci Bluetooth RFCOMM [10].

2.2.2.6 Service Discovery Protocol

Poskytuje aplikacím prostředky pro zjištění, které služby jsou k dispozici a v jakém jsou tyto služby stavu. Tento protokol je důležitý zejména jako soubor služeb, které zjišťují dynamicky se měnící stav signálu zařízení, které mohou být v pohybu. SDP byl navržen s ohledem na tyto specifické vlastnosti Bluetooth sítě.

2.2.2.7 Profiles

Pro každou činnost je využíván jiný tzv. bluetooth profil. Jedná se o soubor instrukcí, pomocí nichž mohou dvě zařízení komunikovat. Profilů existuje celá řada a ne všechny se vyskytují ve všech zařízeních. Profil je specifický pro každou činnost, kterou může dané zařízení vykonávat.

2.2.3 Zabezpečení přenosu dat Bluetooth

Jak vyplývá z předchozích kapitol, o ochranu dat se stará protokol LMP. Bezpečnost u Bluetooth je založena na autentizaci a spárování dvou zařízení. Autentizace je založena na dotazu a odpovědi. Všechny zařízení, které používají Bluetooth dávají na dotázání informaci o třídě zařízení, názvu, seznamu služeb které mohou poskytovat a technické informace kterými jsou výrobce, funkce a verze Bluetooth kterou používají. Každé zařízení může vyhledávat další zařízení s touto technologií.

2.2.3.1 Správa spojení

Protokol LMP nejdříve ověřuje, která zařízení jsou v jeho dosahu. Tato činnost se skládá z několika kroků.

- nejdříve si zařízení přepošlou ověřovací zprávy, v případě že jsou přenesená data v pořádku, tyto zprávy si následně obě zařízení potvrdí.
- poté může dojít k párování dvou zařízení.

Bluetooth do verze 2.0 používá k vytvoření šifrovacího klíče proceduru tzv. párování, kdy je klíč vytvořen na základě krátkého hesla či PINu. Samotná procedura párování se však ukázala jako zranitelná a konstrukcí vhodné zprávy umožňuje útočnickovi párování i bez znalosti PINu. V mnoha jednodušších zařízeních je navíc PIN přednastaven (typicky s hodnotou 0000), což činí útok ještě snadnější. Se získaným šifrovacím klíčem již lze pak snadno pasivně odposlouchávat probíhající komunikaci (s tzv. Bluetooth puškou i na více než 1,5 km). Poznamenejme dále, že slabiny obsahuje i použitý šifrovací algoritmus E0, na jehož prolomení a získání 128bitového klíče je se znalostí dostatečného množství otevřeného textu potřeba jen 2^{38} operací (oproti očekávaným 2^{128} , což odpovídá cca 19 hodinám (na sběr dostatečného množství dat je potřeba 37 hodin) [12].

Bluetooth od verze 2.1 používá párování SSP a má následující režimy:

- přímé párování - tato metoda funguje bez nutnosti spolupráce s uživatelem, může vyžadovat potvrzení párování
- číselné porovnání - tato metoda zobrazí 6místný číselný kód na každém zařízení a uživatel musí porovnat, jestli jsou tato čísla shodná
- přístupový klíč – může být použito mezi zařízeními, z nichž jedno má displej a druhé má tlačítka nebo mezi dvěma zařízeními s klávesnicí, klávesnice slouží k zadání šestimístného kódu
- mimopásmová (OOB)

2.2.3.2 Úroveň zabezpečení

Zabezpečení Bluetooth se vyvíjelo postupně a obsahuje několik úrovní:

- úroveň 1: Neobsahuje žádné zabezpečení, zařízení pracují v tzv. „promiskuitním módu, navzájem si nebrání v navázání spojení. Tato úroveň je podporována pouze ve verzi 2.1 EDR a nižších.
- úroveň 2: Vyžaduje autorizaci, o kterou se stará bezpečnostní manažer, který určuje různou úroveň zabezpečení službám daných zařízení. Různé úrovně zabezpečení dovolují zařízením přístup k určitým službám, podle jejich potřeby. Tuto úroveň podporují všechny verze Bluetooth.
- úroveň 3: Definiuje mechanismy, které jsou volané ještě před vytvořením spojení na fyzické vrstvě. Protokoly této úrovně zajišťují autentizaci a šifrování přenosu. Tato autentizace a šifrování jsou založeny na symetrické kryptografii a využívají sdílený klíč vytvořený při prvním párování těchto zařízení. Podpora této úrovně je možná pouze při spojení dvou zařízení, které mají verzi 2.1 EDR nebo nižší.
- úroveň 4: tato úroveň vynucuje zahájení bezpečnostních procedur pro vytvoření spojení. Je použit protokol SSP, který využívá Diffie-Hellmanův protokol na bázi eliptických křivek, která má sloužit proti Man-in-the-middle útoku. V této úrovni jsou použity stejné mechanismy jako v úrovni 2. Tato úroveň je povinná pro verzi 2.1 EDR.
- úroveň 5: Zajišťuje ochranu přenosu dat při využití AMP. Párování zařízení probíhá pomocí SSP protokolu a výsledný klíč je pomocí HMAC funkce h_2 přeměněn na specializovaný a obecný linkový klíč. Oba tyto klíče jsou následně předány vrstvě PAL (Protocol Adaptation Layer). Tato úroveň je vyžadována ve verzích 3.0 HS a 4.0.

- úroveň 6: Tato úroveň je definován pro Bluetooth 4.0, a je definován pro zařízení s nízkou spotřebou. O zabezpečení na této úrovni se stará bezpečnostní manažer, který poskytuje tři úrovně zabezpečení. Párování zařízení je oproti verzi 3.0 podstatně zjednodušeno a obsahuje pouze výměnu dat na bázi výzva-odpověď. K šifrování je použito šifrování standartu AES s algoritmem CCM a poskytuje jak šifrování, tak autentizaci.

2.2.3.3 *Generování klíčů*

U Bluetooth se používají 128bitové klíče. Tento klíč je využit ke spárování dvou zařízení a je rovněž využit pro vytvoření klíče určeného při přenosu dat. Existuje několik způsobů, jakými jsou klíče používány:

- Unit-Key – je závislý na jednom zařízení, ve kterém je vygenerován. Je měněn pouze ve výjimečných případech a je využíván v zařízeních s malou pamětí.
- Combination-Key – je vytvořen pomocí dvou zařízení, které mají být spárována, jinak má stejnou funkci jako předešlý.
- Master-Key – dočasný klíč, slouží jako náhrada originálního linkového klíče. Tento klíč je využit v případě, kdy je třeba použít jeden klíč pro celou síť.
- Inicialization-Key – je použit pro inicializaci v době kdy ještě není vytvořen žádný linkový klíč [14].

2.2.3.4 *Protokol LP (Legacy Pairing)*

Tento protokol je využíván pouze na úrovni zabezpečení 2 a 3. Při párování je vygenerováno náhodné číslo, které iniciátor spojení přepośle druhému zařízení. Pomocí tohoto čísla a vloženého PINu je vytvořen inicializační klíč. Následně se v každém zařízení vygeneruje náhodné číslo a pomocí těchto čísel a inicializačního klíče je vytvořen kombinační klíč, který si obě strany navzájem přepošlou. Pomocí těchto dvou klíčů se již vytváří linkový klíč, který umožňuje spárování dvou zařízení.

2.2.3.5 *Protokol SSP (Secure Simple Pairing)*

Tento protokol poskytuje tři různé druhy autentizace, které jsou zvoleny podle toho jaká zařízení spolu mají komunikovat. Tento protokol využívá asymetrické kryptografie na bázi asymetrických křivek, díky čemuž je odolnější proti útoku „Man-in-the-middle“. Klíče jsou vytvořeny pomocí algoritmu Diffie-Hellman. Tyto klíče jsou následně používána při dalších spojeních těchto zařízení.

2.3 ZigBee

2.3.1 Specifikace

Tato komunikační technologie je popsána standardem IEEE 802.15.4. Patří do skupiny bezdrátových sítí PAN stejně jako technologie Bluetooth, která se využívá hlavně ve spotřební elektronice, ale existují oblasti pro které technologie Bluetooth není vhodná a právě tady nalézá své uplatnění technologie ZigBee. Pracuje pomocí rádiového rozhraní na frekvenci 868 MHz, 902–928 MHz a 2,4 GHz. Přenosová rychlost zařízení je až 250kbit/s, což znamená, že není vhodná pro přenos větších objemů dat. Tato technologie se uplatňuje především v průmyslových aplikacích a v senzorových sítích. Díky použití multiskokového ad-hoc směrování umožňuje komunikaci i na větší vzdálenosti bez přímé radiové viditelnosti jednotlivých zařízení, což znamená, že dvě zařízení které nejsou v přímém dosahu, mohou komunikovat pomocí třetího zařízení, které je s nimi zapojeno v síti a má dosah na obě tyto zařízení. Tato technologie byla vytvořena ZigBee aliancí, kterou tvoří přes šedesát soukromých firem. Technologie ZigBee je navržena jako jednoduchá a flexibilní technologie zejména pro tvorbu rozsáhlejších bezdrátových sítí, u nichž není požadován přenos velkého objemu dat. Tuto technologii vyznačují tři standardy přenosu dat.

- periodicky se opakující přenosy (např. přenosy dat ze snímačů a čidel)
- nepravidelné přenosy (např. přenos informace o stisknutí tlačítka na zařízení)
- periodické přenosy s požadavkem na malé zpoždění (spotřební elektronika)

U této technologie se klade důraz na maximální jednoduchost implementace protokolů. Protokol této implementace se skládá ze tří vrstev. Obsahuje fyzickou, síťovou a aplikační vrstvu. Fyzická vrstva popisuje přístup k přenosovému zařízení. Na síťové vrstvě probíhá realizace připojení k síti, správa směrování paketů a zabezpečení přenášených dat. Aplikační vrstva zajišťuje služby jednotlivých aplikací. Zařízení standardu ZigBee se dělí na dva druhy:

- FFD(Full functional device) – tato zařízení implementují plný protokolový rámec
- RFD(Reduced functional device) - implementují pouze nezbytné protokolové knihovny z důvodu maximálního omezení hardwarové náročnosti, mohou pracovat pouze jako koncová zařízení, mohou komunikovat pouze s koordinátorem sítě a jsou omezeny na hvězdicové uspořádání topologie

2.3.2 Zabezpečení přenosu dat

Jako základní zabezpečení ZigBee se používá symetrické šifrování AES-CCM (Advanced Encryption Standard) s klíčem o délce 128 bitů. Toto zabezpečení je implementováno v síťové přenosové vrstvě. Proto aby každé zařízení mělo svůj klíč se používají tři metody.

- klíče jsou nastavené od výrobce
- klíče jsou přeposlány z důvěrného centra Security Service Provider
- vytvoření klíče pomocí PINu

U technologie ZigBee je kladen důraz na cenu a jednoduchost zařízení. Veškerá bezpečnost je cílena pouze na kryptografickou ochranu rámců, v případě že má někdo fyzický přístup k tomuto zařízení, dostane se i ke klíčům pro šifrování dat. Tyto klíče jsou rozděleny na tři druhy:

- Master key – tento klíč slouží pro odvození dalších hesel, je nastavený od výrobce
- Link Key – klíč sloužící k šifrování dat
- Network key – klíč sloužící k přihlašování do sítě

2.4 Wi-Fi

2.4.1 Specifikace

Technologii wi-fi popisuje standart IEEE 802.11. Tato technologie využívá bezlicenčního pásma 2,4 GHz norma IEEE 802.11b/g/n případně pásma 5Ghz IEEE 802.11a. Norma IEEE 802.11b byla schválena v roce 1999, přenosová rychlost je až 11Mbit/s, využívá pásmo 2.4GHz. Ve volném prostředí může mít dosah až 12km. Toto zařízení může trpět rušením od jiných zařízení, které pracují na stejných frekvencích jako jsou mikrovlnné trouby, případně zařízení Bluetooth, bezdrátové telefony a další zařízení. Norma IEEE 802.11g vznikla v roce 2003, má zvýšenou propustnost až na 54Mbit/s. Pracuje na stejné frekvenci 2.4 GHz jako norma IEEE 802.11b ale používá odlišnou modulaci tzv. „ortogonální frekvenční multiplex“ OFDM. Tuto modulaci používá také norma IEEE 802.11a, která pracuje na frekvenci 5GHz. Díky tomu lze využívat vyšších vysílacích výkonů než u ostatních norem pracujících na frekvenci 2.4Ghz. Kromě toho existují další normy, norma 802.11d se využívá v některých zemích, zařízení vysílá v broadcastu ISO kód země, a podle toho se přizpůsobí výkonově normám v jednotlivých zemích. Norma IEEE 802.11e má další vylepšení pro aplikace citlivé na kvalitu připojení, jako třeba metodu mnohonásobného pří-

stupu s nasloucháním nosné a zabráněním kolizím, tzv. QoS(Quality of service), nebo podporu správy napájení.

2.4.2 Možnosti zabezpečení Wi-Fi

Bezdrátová zařízení se prodávají s nastavením bez jakéhokoliv zabezpečení, aby po zakoupení fungovala ihned po zapojení do sítě. Vývoj různých zabezpečení probíhal postupně spolu s rozšiřováním těchto sítí. Zabezpečení těchto sítí se dá rozdělit do dvou základních problémů.

- zabezpečení přenášených dat proti odposlechu
- přihlašování do sítě

Bezdrátová zařízení mají omezený dosah, ale za použití směrové antény se lze k síti přihlásit až na vzdálenost několika kilometrů. Existuje několik možností jak zabezpečit wi-fi síť. Tyto možnosti si popíšeme v následujících odstavcích.

2.4.2.1 Zablokování SSID

Každá wi-fi síť vysílá svoje SSID (Service Set Identifier), což je řetězec ASCII znaků, pomocí kterého dochází ke spojení jednotlivých síťových adaptérů. Jinými slovy touto zprávou dává síť o sobě vědět ostatním adaptérům. Pokud je zablokování SSID jedinou ochranou wi-fi sítě, je tato ochrana nedostačující, protože při přihlašování do sítě je třeba odeslat SSID jako nešifrovaný text, i když přenosy na této síti jsou jinak šifrované. Moderní přístupové body nabízejí možnost tvorby virtuálních SSID, kdy je možné vytvořit na jednom přístupovém bodu několik logických s různým zabezpečením.

2.4.2.2 Kontrola MAC adres

U každého přístupového bodu je možné nastavit kontrolu připojovaných zařízení pomocí MAC adres. Toto zabezpečení spočívá v tom, že přístupový bod obsahuje seznam MAC adres zařízení, kterým povolí přístup do sítě. Tuto ochranu lze obejít změnou MAC adresy u zařízení. Na tuto změnu adresy síťové karty existuje volně dostupný software.

2.4.2.3 WEP

Protokol „Wired Equivalent Privacy“ byl výchozím šifrovacím protokolem, jenž byl poprvé uveden v roce 1999 ve standardu IEEE 802.11. Protokol je založen na principu šifrovacího protokolu RC4 a používá šifrovací klíč o délce 40 nebo 104 bitů, kombinovaným s 24bitovým inicializačním vektorem (IV) pro šifrování textové zprávy M a jejího kontrolní-

ho součtu – ICV (Integrity Check Value). Šifrovaná zpráva C byla proto určena pomocí následujícího vzorce:

$$C = [M \parallel ICV(M)] + [RC4(K \parallel IV)]$$

kde \parallel představuje operátor zřetězení a $+$ operátor XOR. Klíčem k bezpečnosti WEP je samozřejmě inicializační vektor, takže k udržení přiměřené úrovně zabezpečení a zmenšení možnosti odhalení by měl být IV zvětšen pro každý paket tak, aby se následné pakety šifrovaly odlišnými klíči. IV se bohužel pro bezpečnost protokolu WEP přenáší jako nešifrovaný text a standard 802.11 nenařizuje zvyšování IV, čímž ponechává toto bezpečnostní opatření na volbě jednotlivých implementací bezdrátových terminálů (přístupových bodech nebo bezdrátových kartách) [13].

Protokol WEP má dvě slabá místa v protokolu RC4- invariance a známé útoky na inicializační vektor. Oba útoky se spoléhají na skutečnost, že u určitých hodnot klíčů je možné, aby „u-bity“ v počátečních bajtech keystreamu (proudu klíčů) závisely pouze na několika bitech šifrovacího klíče (ačkoli běžně je asi padesátiprocentní pravděpodobnost, že každý proud klíčů bude odlišný od předchozího proudu). Jelikož šifrovací klíč se sestává zřetězováním tajného klíče s IV, určité hodnoty IV poskytují slabé klíče [13]. Na prolomení tohoto zabezpečení existují nástroje jako je Aircrack, Aircrack-ng nebo WepLab.

Další slabinou je kontrola integrity dat CRC32, tento algoritmus nebyl nikdy považován za zcela bezpečný. CRC32 je hashovací funkce používaná k detekci chyb během přenosu dat. Kontrolní součet bývá odeslán či ukládán společně s daty, při jejichž přenosu nebo uchování by mohlo dojít k chybě. Po převzetí dat je znovu spočítán a potom je porovnán s původním kontrolním součtem. Pokud nedošlo při přenosu dat k chybě, musí oba kontrolní součty odpovídat. Tento algoritmus se hodí jako ochrana dat proti poruchám, ale nehodí se jako ochrana dat proti úmyslné změně dat. CRC32 používá k výpočtu kontrolního součtu operátor „XOR“ což znamená, že vybraný bajt v šifrované zprávě odpovídá bajtu v původní zprávě. „Pokud se tedy poslední bajt šifrované zprávy oddělí, zpráva se sice poruší, ale rovněž tím lze uhodnout hodnotu odpovídajícího nešifrovaného bajtu a podle toho šifrovanou zprávu opravit“ [13]. K tomuto útoku byl vytvořen nástroj s názvem chopChop, který umožňuje dešifrovat pakety i bez znalosti šifrovacího klíče. Protokol WEP má tak několik slabin, a dnes se jeho používání nedoporučuje.

2.4.2.4 WPA

Toto zabezpečení je zpětně kompatibilní se zabezpečením WEP. Bylo vytvořeno jako reakce na prolomení zabezpečení WEP v roce 2001. Cílem bylo nejen nahradit zabezpečení WEP, ale také umožnit využívat toto zabezpečení starými zařízeními pomocí aktualizace software. Využívá WEP klíče, které jsou ale dynamicky bezpečným způsobem měněny. Autentizace je prováděna pomocí PSK (Pre-Shared-Key) nebo pomocí přihlašovacího jména a hesla (RADIUS server). WPA používá šifru RC4. Pro odstranění slabých míst této šifry byl vyvinut protokol TKIP. Tento protokol odstranil problém s inicializačními vektory a zavedl dynamickou správu šifrovacích klíčů. Tyto klíče jsou na rozdíl od WEP přenášeny i během komunikace a nejen na začátku přenosu jako tomu bylo u WEP. Toto zabezpečení funguje tak, že na straně klienta je spuštěn program zajišťující autentizaci klienta a správu šifrovacích klíčů pomocí TKIP. WPA používá 128bitový šifrovací klíč a 48bitový inicializační vektor. Díky tomu je odolnější proti útokům než WEP. Dále tento protokol vylepšuje kontrolu integrity dat. Používá k tomu metodu MAC (Message Authentication Code). Pokud je WPA použito společně s TKIP, je možné toto zabezpečení prolomit. Autentizace se provádí pomocí přihlašovacího jména a hesla, ale většina operačních systémů umožňuje tyto informace uložit, což při odcizení počítače znamená bezpečnostní hrozbu. Další ochranou WPA je použití funkce PBKDF2 (Password-Based Key Derivation Function), pro odvozování klíčů. Nebezpečím je zadání slabého hesla, které neodolá útoku hrubou silou.

2.4.2.5 WPA2

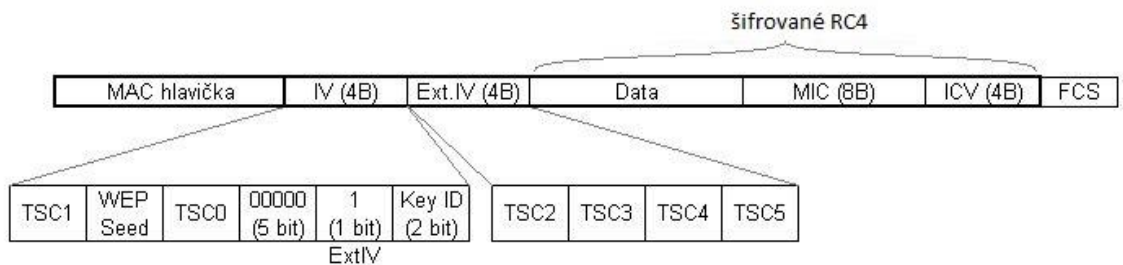
Standart IEEE 802.11i byl schválen v roce 2004. Odděluje autentizaci uživatele od vynucování integrity a soukromí zprávy, čímž poskytuje dostatečnou bezpečnost pro veškeré aplikace těchto sítí. Tato architektura nese označení RSN (Robust Security Network). WPA2 používá autentizaci 802.1X silnou distribuci klíčů a nové mechanismy

k zajištění integrity a soukromí. WPA2 používá tuto hierarchii klíčů:

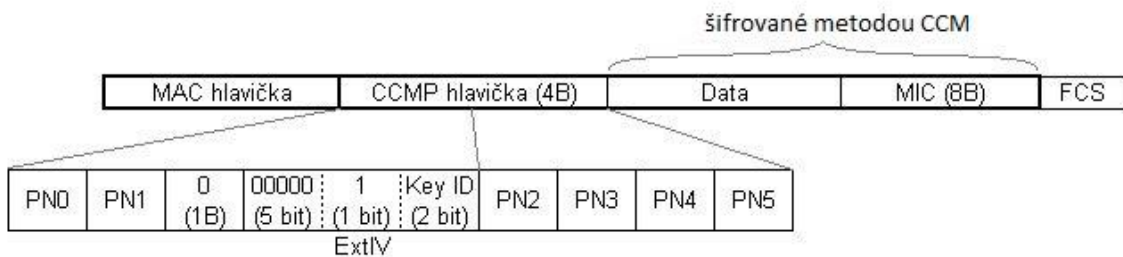
- Pairwise Master Key (PMK) - hlavní párový klíč, což je hlavní klíč mezi přístupovým bodem a každou stanicí, rozeznání tohoto klíče se dokazuje pomocí čtyřcestného EAPOL(802.1x)
- Pairwise Transient Key (PTK) - přechodný párový klíč, což je klíč odvozený z PMK, použije se v daném připojení (session) na vytváření klíčů pro šifrování a autentifikaci

- Group Transient Key (GTK) – (přechodný skupinový klíč) určený pro všechny stanice na dešifrování broadcast komunikace
- EAPOL-Key Encryption Key (KEK) a EAPOL-Key Confirmation Key (KCK) – klíče pro přenos klíčů přes EAPOL (klíč na šifrování klíče; klíč na potvrzování klíče) – derivované z PTK
- Temporal Key (TK) – (dočasný klíč) klíče pro šifrování a zabezpečení integrity jednoho datového rámce – derivované z PTK a počítadel rámců.

Nově je přidán algoritmus CCMP založený na AES, který nahrazuje TKIP algoritmus. Na obrázku 4. Je znázorněno zapouzdření šifrování dat pomocí TKIP, a na obrázku 5. metodou CCM.



Obrázek 5: Zapouzdření TKIP [14]



Obrázek 6: Zapouzdření CCMP [14]

Norma WPA2 vyžaduje zabezpečení přenosu dat pomocí CCMP protokolu (obr 5). Tento protokol používá 128bitové AES, konkrétně šifru Rijndael, což je bloková šifra, která používá 128bitový klíč a 128bitovou délku bloku. MIC(Message integrity code) je integritní kód vypočítaný pomocí CBC-MAC. PN – což je číslo paketu se spolu s poli z MAC-hlavičky použije na vytvoření jednorázové hodnoty pro zabezpečení integrity dat v bloku šifrovaném metodou CCM.

2.5 Síť GPS

GPS (Global Positioning System) je satelitní družicový systém, který provozuje Ministerstvo obrany Spojených států amerických. Přesnost tohoto systému je v desítkách metrů, některými dalšími metodami lze tuto přesnost ještě zvýšit. Část služeb tohoto systému je k dispozici civilnímu obyvatelstvu. Tento systém je dnes běžně využíván například v mobilních telefonech nebo navigačních systémech. Systém GPS byl vytvořen v době, kdy nebyly dané požadavky na utajení dat. Tento systém nepoužívá žádnou kryptografickou ochranu, je zmíněn jenom pro úplnost.

2.6 Síť GSM

2.6.1 Specifikace

Síť GSM (Globální systém pro mobilní komunikaci) je nejrozšířenější standart používaný pro komunikaci mobilních telefonů na světě. V současnosti jej používá více než 5 miliard účastníků na světě. Jméno systému GSM pochází z názvu pracovní skupiny „Groupe Spécial Mobile“, která navrhla několik prvních verzí tohoto standardu. Od svého vzniku prošel systém GSM mnoha modernizacemi a dnes je v provozu již několikátá verze tohoto systému.

2.6.2 NMT, AMPS, TACS, C450 - síť 1. generace

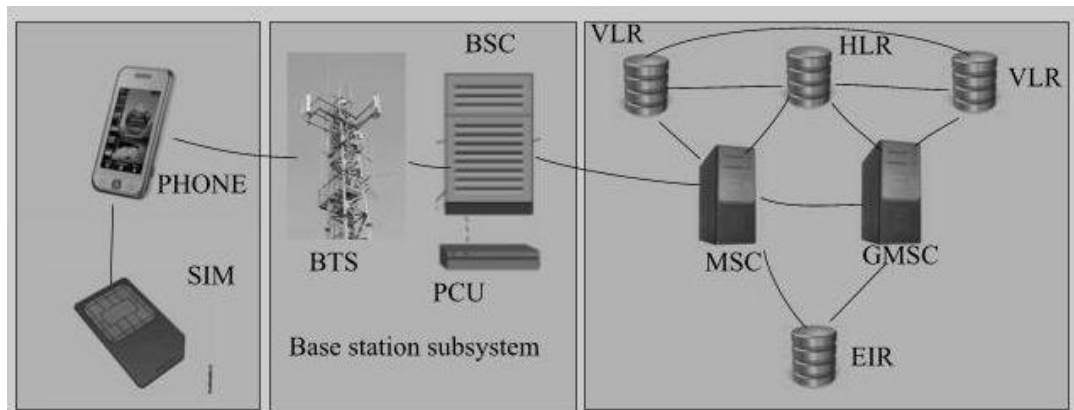
Tyto sítě byly uvedeny do provozu v 80. letech 20. století. V té době se předpokládalo, že mobilní telefony bude používat jenom malá část populace. Telefony v té době byly velké a těžké, a byly montovány zejména do automobilů. Signál těchto sítí pokrýval jenom největší města. Jednalo se o nadstandartní službu a tyto služby byly velmi drahé. Tyto sítě byly využívány jen v nejvyspělejších zemích a jednotlivé země mezi sebou neměly roamingové smlouvy. Tento systém využíval pásmo 450MHz. Tato síť je označovaná jako síť 1. generace. Pro rozšíření kapacity později začala tato síť používat frekvenci 900MHz. Technické principy byly definovány v roce 1973, specifikace pro základnové stanice v roce 1977. Tento standart byl otevřený a tak se brzy začal rychle rozšiřovat a začala se snižovat cena mobilních komunikačních služeb. V České republice byla tato síť provozována společností Eurotel v letech 1991-2006. Na frekvenci 450MHz má tato síť nevýhodu v malé kapacitě pro počet účastníků, ale její výhoda spočívá v dobrém šíření signálu zejména v kopcovitém terénu. NMT používá duplexní provoz. Velkou nevýhodou tohoto systému je že analogový a první jeho verze neumožňovala žádnou kryptografickou ochranu dat. Hovory mohl v té

době kdokoliv odposlouchávat, kdo měl přijímač příslušného pásma. První snahy k ochraně odposlouchávání těchto telefonů vedly k vypouštění příslušných kmitočtových pásem z rozsahu komunikačních přijímačů. Pozdější verze NMT standartu umožňovaly volitelné analogové utajení založené na dvoupásmové inverzi zvukových kmitočtů. Utajení lze použít, pokud jej podporují mobilní telefony obou účastníků nebo aspoň jeden mobilní telefon a základnová stanice. Síť AMPS (Advanced Mobile Phone System) byla vyvinuta společností Bell's labs. Je to další analogový mobilní telefonní systém poprvé představený v roce 1986 v USA. Mobilní telefony pro tento systém zpočátku vyvinula firma Motorola. Na rozdíl od předešlých telefonů sítě NMT byly tyto telefony menší a daly se už normálně přenášet. Tento systém trpěl ještě mnoha nedostatky a stejně jako systém NMT neměl žádnou ochranu proti odposlechu. Navíc měl jeden velký nedostatek, a sice že při spojování telefonního hovoru bylo přeposíláno ESN (identifikační číslo účastníka sítě) a mohlo být zachyceno cizí osobou. Toto ESN bylo využíváno při vyúčtování hovorů a telefonní přístroj s tímto ESN mohl být dále naklonován.

2.6.3 GSM síť 2. generace

Digitální systémy mobilní telefonie se nazývají síť 2. generace. V roce 1989 Evropská telekomunikační standartizační instituce ETSI oficiálně definovala GSM jako nový mezinárodní digitální telekomunikační buňkový standard a začala dohlížet na jeho vývoj. První specifikace byla dokončena v roce 1991. Síť GSM pracují na frekvencích 900MHz, 1800MHz, případně na frekvenci 1900MHz. Použití digitálního přenosu způsobilo rychlý rozvoj mobilních technologií. To umožnilo použití dalších vymožeností, jako jsou SMS, datové přenosy, zobrazování čísel volajících, přesměrování hovorů, hlasových schránek a dalších služeb. Rovněž se u sítí 2G zvýšila bezpečnost přenášených dat. Kódování zvukového signálu a převod do digitální podoby umožnilo zvýšení přenosové kapacity, na přenos kódovaných informací je potřeba jen zlomek času potřebného k hovoru. Systém GSM byl navržen tak, aby ověřoval uživatele za použití šifrování. Tento systém se začal vyvíjet v době, kdy si firmy začaly uvědomovat nutnost ochrany přenášených dat, ale ještě nebyly na dostatečné úrovni. Proto má systém GSM také bezpečnostní slabiny. Bezpečnostní model nabízí důvěrnost a autentičnost, ale omezené autorizační schopnosti. Jednou z klíčových vlastností bezpečnosti GSM je SIM (Subscriber Identity Module). Je to čipová karta, která obsahuje informace potřebné k přihlášení uživatele do sítě a může na ní být uložen telefonní seznam a SMS. Karta se dá z mobilního telefonu vyndat a vložit do jiného telefonu, což umožňuje účastníkovi jednoduchou změnu telefonu bez změny telefonního čísla.

Další možností je použití více SIM karet různých operátorů v jednom telefonu. Struktura GSM sítě je znázorněna na obr. 6.



Obrázek 7 : schéma GSM sítě

Síť GSM lze po technické stránce rozdělit na čtyři systémy:

- Mobile station – mobilní stanice, skládá se z telefonu a sim karty
- Base Station Subsystem – Subsystem základnových stanic
- The Network Switching Subsystem - síťový a spojovací subsystem
- Operating and Maintenance Subsystem – subsystem řízení a údržby

Tyto čtyři subsystemy jsou vzájemně propojeny a každý z nich plní svoji funkci. Systém GSM je dostatečně flexibilní aby do něj mohly být přidávány další součásti. To se také postupně děje a původní systém byl postupně rozšiřován o další technologie, jako jsou třeba datové přenosy. Digitální systém je v porovnání s analogovým bezpečnější, je možné přenášet data šifrovat a odposlech tohoto systému je téměř nemožný. Systém GSM obsahuje čtyři způsoby zabezpečení proti zneužití dat:

- použití SIM karty
- anonymitu TMSI (Temporary Mobile Subscriber Identity)
- ověření totožnosti
- ochranu signalizačních a hovorových dat šifrováním

K tomu používá tyto algoritmy:

- A3 - pro ověření totožnosti účastníka (může být definován operátorem)
- A5 - pro šifrování a dešifrování dat (algoritmus je normalizovaný pro všechny sítě GSM)
- A8 - pro generování šifrovacího klíče (může být definován operátorem) [14]

Přestože A3 a A8 nejsou normovány, vznikl jeden normovaný algoritmus označovaný jako COMP128, který plní funkce A3 a A8. Tento algoritmus vytvořila firma „Deutches Telecom“, a je využíván převážnou většinou operátorů.

2.6.4 SIM karta (subscriber identity module)

Karta SIM chráněna číslem PIN. Toto číslo bývá standartně čtyřmístné, a po zapnutí telefonu jej musí uživatel zadat na klávesnici. Poté je PIN porovnán telefonem s číslem na kartě SIM. Pokud uživatel zadá PIN třikrát špatně, telefon je následně zablokován a je možné jej odemknout zadáním čísla PUK (Personal Unblocking Key). Pokud je špatně zadáno číslo PUK desetkrát po sobě, SIM karta je zničena. „Mobilní stanice komunikuje se systémem pouze v případě, je-li do ní vložena SIM karta. Výjimku tvoří případy tísňového volání. Karta SIM slouží k identifikaci uživatele a lze ji zasunout do libovolného (např. vypůjčeného) mobilního telefonu GSM. Obsahuje čip s mikroprocesorem a paměťmi RAM a ROM, ve kterých jsou uloženy důležité informace uživatele“ [14]. Karta SIM obsahuje tyto údaje:

- PIN
- PUK
- IMSI(International Mobile Subscriber Identity), číslo, které obsahuje identifikační údaje mobilního telefonu
- tajný ověřovací klíč K_i a algoritmy A3 a A8
- telefonní seznam, zprávy SMS

2.6.5 Ověření totožnosti v síti GSM

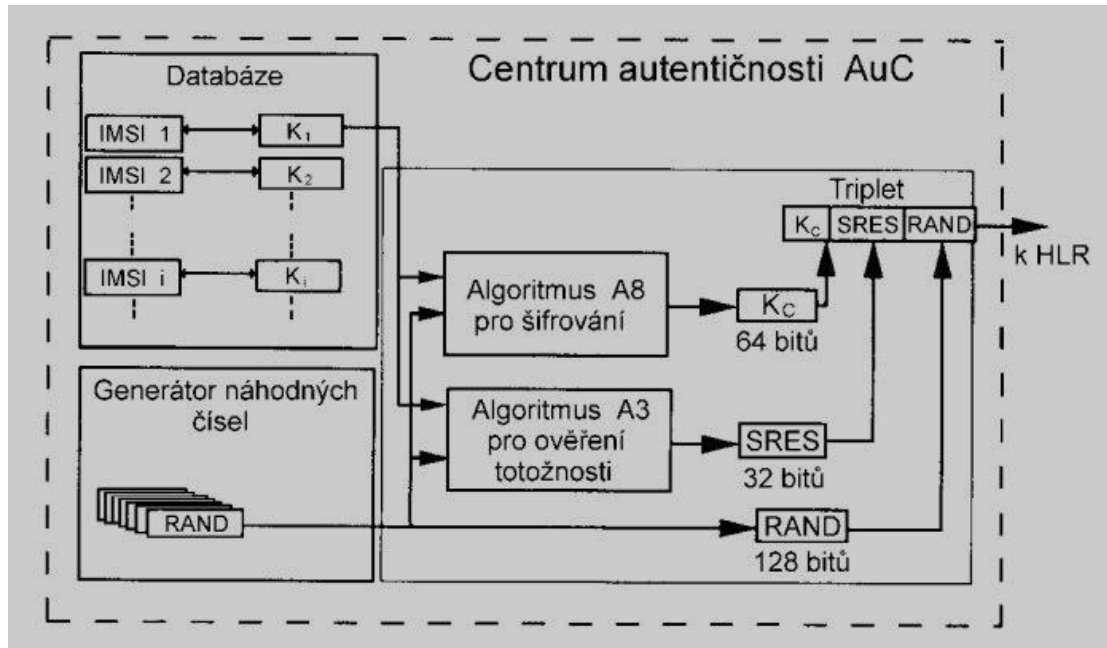
„Každý účastník je v síti GSM jednoznačně identifikován číslem IMSI. Aby nemusel toto číslo posílat přes rádiové rozhraní při každé žádosti o nějakou službu, přiřadí mu systém tzv. dočasnou identifikaci TMSI (Temporary Mobile Subscriber Identification). Číslo TMSI je uloženo na SIM kartě a v registru VLR mobilní ústředny. Pokud se účastník s mobilní stanicí přesune na území pod kontrolou jiné ústředny, je mu novou ústřednou zasláno nové číslo TMSI a předchozí číslo je zrušeno jak v SIMkartě, tak i ve VLR předchozí ústředny. Takovým anonymním způsobem se účastník pohybuje v síti GSM. Pouze v případech, kdy se účastník hlásí do systému po zapnutí mobilní stanice, zasílá MS do ústředny identifikaci IMSI. Ihned poté je však do mobilní stanice zaslána prozatímní identifikace TMSI, pomocí které již může účastník se systémem dále komunikovat, tj. žádat o služby, atd.“ [14].

„Ověření totožnosti účastníka může být provedeno až tehdy, když systém zná IMSI (TMSI), neboť při znalosti této identifikace může systém použít i další tajné informace potřebné k výpočtům. Jedná se o okamžitou kontrolu totožnosti účastníka, která se provádí technikou nazývanou „Výzva a odezva“ (Challenge and Response). V AuC je generováno 128bitové náhodné číslo RAND, které je zasláno k MS, kde se z něj vypočítá odezva SRES (Signed Response) jež se zasílá zpět. Obdobný výpočet se provádí i v AuC a oba výsledky se srovnávají. Přenos signálů probíhá mezi MS a AuC-MSC, BSS je pouze průchozí. V případě shody čísel SRES je účastníkovi povolen přístup k systému, v opačném případě je přístup odmítnut“ [14]. Schéma ověření totožnosti je na obrázku 7.

„Odezva SRES se počítá pomocí účastnického (individuálního) tajného ověřovacího klíče a náhodného čísla RAND, užitím algoritmu A3. Výsledkem tohoto procesu je 32bitové číslo SRES“ [14].

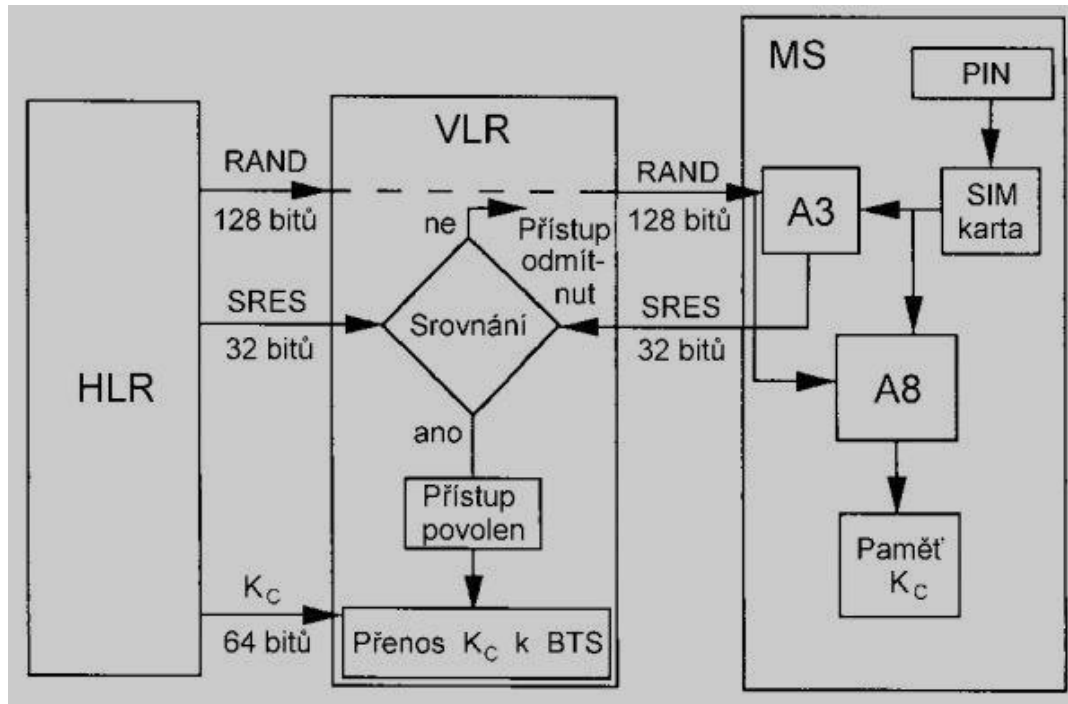
„Současně s generací čísla RAND se v AuC počítá kromě odezvy SRES i tzv. šifrovací klíč K_c , který bude při uskutečnění hovoru použit k šifrování přenášených dat. Šifrovací klíč K_c se počítá z tajného ověřovacího klíče K_i a náhodného čísla RAND pomocí algoritmu A8. Jeho délka je 64 bitů“ [14].

Trojice čísel RAND, SRES a se nazývá triplet a lze ji generovat teprve na základě znalosti identifikace účastníka pomocí IMSI nebo TMSI. Po výpočtu je triplet uložen do HLR. Při každé další komunikaci účastníka se systémem se generuje nový triplet [14]. Princip generace tripletu je znázorněna na obr. 7.



Obrázek 8 : Generace tripletu [14]

Celý proces ověření totožnosti začíná požadavkem VLR-MSR zaslaným do AuC na generaci tripletu. Požadavek z VLR může vzejít na základě žádosti účastníka o komunikaci nebo na základě žádosti ústředny, která s účastníkem chce komunikovat (volá ho jiný účastník). Triplet se generuje před začátkem komunikace pro oba účastníky sice stejným způsobem ale s různými náhodnými čísly RAND a samozřejmě různými ověřovacími klíči každého účastníka, takže vypočítaná čísla jsou zcela rozdílná. Po generaci je triplet uložen do HLR. Z něj se na pokyn VLR zasílá číslo RAND k účastníkovi a odezva SRES spolu s šifrovacím klíčem se zasílá do VLR. Ve VLR se provádí srovnání obou čísel SRES a v případě shody je šifrovací klíč zaslán do základnové stanice BTS. Lokační registr VLR tedy zahajuje ověřovací proces a rovněž kontroluje i jeho výsledek [14].



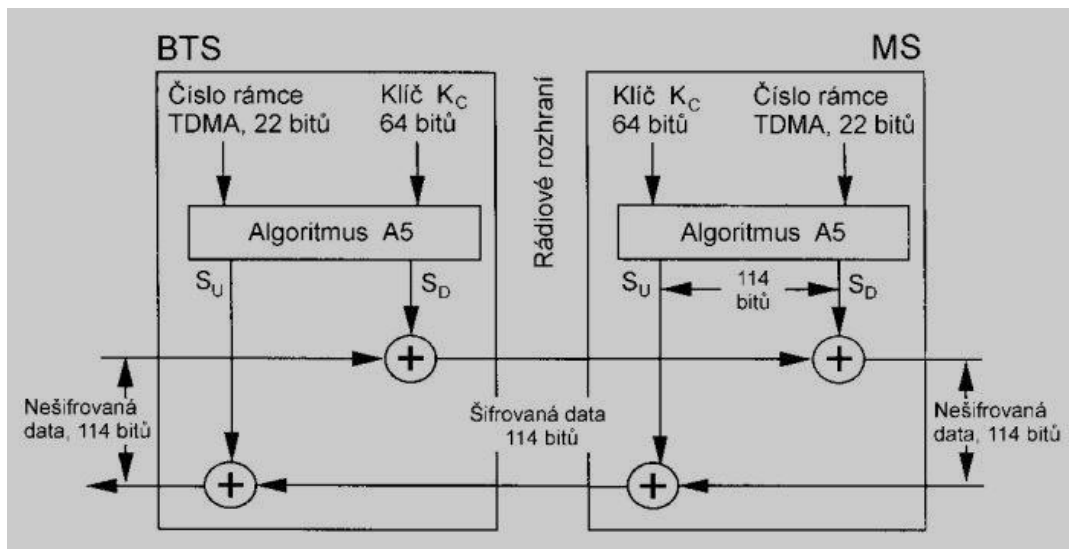
Obrázek 8 : Ověření totožnosti uživatele GSM[14]

2.6.6 Šifrování přenášených dat v síti GSM

Šifrovacím procesem se zabezpečují data proti zneužití. V síti GSM se signál zabezpečuje šifrováním pouze při přenosu v bezdrátovém prostředí. „Počet transformačních operací je určen šifrovacím klíčem K_c “ [14]. Jak je zřejmé z předchozí kapitoly, o šifrovacím algoritmu rozhoduje operátor sítě GSM. „Blok, ve kterém je signál šifrován, je zařazen mezi blok prokládání (Interleaving) a modulátor. Dešifrování se provádí mezi demodulátorem a blokem zpětného prokládání (Deinterleaving). Šifrování se provádí pouze nad hovorovými nebo datovými signály účastníka a nad některými signálními signály, tedy nad bity každého burstu. Ostatní bity burstu se nešifrují (ekvalizační sekvence, okrajové bity, atd.)“ [14]. Šifrovací klíč se mění při každé komunikaci. Šifrovací proces je znázorněn na obr. 9.

„Šifrovací algoritmus je synchronní s rámcí TDMA a pouze nepatrně komplikuje složitost zpracování signálu jak v MS, tak i v BTS. Z čísla TDMA rámce a šifrovacího klíče jsou pomocí algoritmu A5 generována v MS i BTS současně dvě 114 bitová slova S_u a S_d , jedno pro uplink a druhé pro downlink. Jednotlivé bity těchto slov jsou ve sčítacích obvodech modulo 2 sčítány se 114 bity každého burstu účastnického signálu. Uvažujeme-li např. přenos na uplinku, potom při šifrování dat v MS pomocí slova S_u , je tím stejným slovem signál v BTS dešifrován. Poněvadž každý následující rámeček má jiné číslo, mění se

s každým rámcem i generovaná slova S_u a S_d . Pokud je během šifrování signálu proveden handover, šifrovací klíč K_c se nemění“ [14].



Obrázek 9 : Šifrovací proces [14]

2.6.7 Přenos dat v síti GSM

2.6.7.1 TCH

Mobilní síť GSM byla původně navržena pro přenos telefonních hovorů, ale s vývojem techniky se postupně začaly do sítě implementovat další technologie pro přenos dat. Tyto technologie jsou dnes využívány ve velké míře a objem dat přenášených pomocí těchto technologií je dnes větší než objem dat telefonních hovorů. Původní technologie GSM umožňovala i datové přenosy, které byly přenášeny stejným způsobem jako data telefonního hovoru. Vzhledem k tomu že tento způsob je v principu náchylný k chybám, takto přenášená data musela být mnohem lépe zabezpečena než telefonní hovory. „Pro přenos datových signálů existuje pět různých datových kanálů s označením TCH/F9.6, TCH/F4.8, TCH/F2.4, TCH/H4.8, a TCH/H2.4, které používají odlišné způsoby kanálového kódování i prokládání. Písmena TCH (Traffic CHannel) označují provozní kanál, písmeno za lomítkem F (Full-rate) resp. H (Half-rate) značí přenos s plnou nebo poloviční rychlostí a desetinné číslo udává přenosovou rychlost signálu v kbit/s“ [14]. Tato technologie má ovšem malou rychlost, data jsou rozdělována do jednotlivých burstů, poté jsou zašifrována a přenesena podobně jako telefonní hovor. Způsob šifrování dat byly u operátorů rozdílné. Ne-

výhodou tohoto způsobu přenosu je nízká přenosová rychlost. Pohybuje se v jednotkách kbit/s.

2.6.7.2 HSCSD (*High Speed Circuit Switched Data*)

Využívá stejnou technologii jako TCH, ale ke zvýšení přenosové rychlosti tato technologie využívá možnosti použití více timeslotů najednou. Podle toho s kolika timesloty dokáže tato technologie pracovat najednou se, rozděluje do osmnácti tříd. Pro implementaci tohoto systému je nutná pouze úprava software v síti. Tato technologie má nevýhodu, že pokud nejsou v síti volné timesloty, přenosová rychlost klesá na nulu. Přednost v síti mají hlasové přenosy.

2.6.7.3 GPRS (*General Packet Radio Service*)

Další technologií, která umožňuje přenos dat je GPRS. Teoretická přenosová rychlost dosahuje až 171,2kbit/s. Tato technologie používá paketový přenos dat a protokol IP. Umožňuje tedy připojení do sítě internet. Systém GSM musel být pro použití této technologie doplněn o další součásti. Přenášená data nejsou posílána do telefonní sítě, ale do jednotky IWU, která je uzpůsobena jako běžná počítačová síť. Tato síť používá čtyři kódovací systémy CS1 až CS4. Nejlépe zabezpečený proti chybám je systém CS1, má však nejnižší přenosovou rychlost. Nejhůře je zabezpečení CS4, právě u tohoto systému je možný přenos dat rychlostí až 171,2kbit/s.

„Technologie GPRS nabízí alternativu k právě popsanému mechanismu fungování na principu přepojování okruhů. Základní charakteristikou, kterou má GPRS (General Packet Radio Service) ostatně již ve svém názvu, je jeho fungování na principu přepojování paketů. Nejlépe je si představit, že GPRS je zcela nová síť, doslova „přeložená“ přes existující mobilní síť GSM, a využívající pouze systém základnových stanic (stanic BTS) k tomu, aby mohla komunikovat s mobilními terminály v dosahu příslušných základnových stanic, prostřednictvím frekvencí které jsou pro příslušnou GSM síť vyhrazeny. „Právě frekvenční kanály, rozdělené pomocí časového multiplexu do jednotlivých slotů, jsou přitom nejvzácnějším kapacitním zdrojem GSM sítí. V praxi, při podpoře GPRS přenosů, s nimi GSM síť hospodář tak, že nejprve přidělí požadovaný počet slotů pro hlasové hovory a pro datové přenosy na principu přepojování okruhů (CSD či HSCSD). Teprve ze zbývajících slotů, pokud vůbec nějaké zbývají, GSM síť uspokojuje momentální požadavky GPRS přenosů. To má významné praktické důsledky pro samotné GPRS. Na jedné straně to efektivně využívá dostupnou přenosovou kapacitou (neplýtvá s ní), což následně vychá-

zí příznivě i po ekonomické stránce. Na druhou stranu to ale znamená nezaručený (negarantovaný) způsob fungování - pokud v daném okamžiku nejsou v dané buňce k dispozici žádné volné sloty, přenosová rychlost GPRS klesá na nulu“ [15].

2.6.7.4 EDGE (Enhanced Data Rates for GSM Evolution)

Pracuje podobně jako předchozí technologie, ale používá modulaci 8PSK(Eight Phase Shift Keying), na rozdíl od předchozích technologií využívajících modulaci GMSK(Gaussian minimum-shift keying). To vede ke zvýšení přenosové rychlosti na 384kbit/s. Systém GSM musel být pro tuto technologii doplněn o další části.

2.7 Sítě 3G – UMTS , HSDPA, HSUPA, HSPA+, LTE

Sítě 3G jsou dalším vývojovým stupněm sítí GSM. O specifikaci tohoto systému se stará sdružení firem a institucí 3GPP (3rd Generation Partnership Project). Specifikace 3G je volně dostupná, její bezpečnost je založená na kvalitních algoritmech a ne na jejich utajování. Sítě 3G dnes fungují ve většině zemí zároveň se sítěmi GSM druhé generace. Účastník v síti GSM, který má aktivovaný datový tarif a telefon s podporou UMTS, bude v místech, kde je dostupný UMTS signál, automaticky UMTS používat. Tato technologie disponuje vyšší přenosovou rychlostí a její využití je zaměřeno více na datové přenosy, než na telefonní hovory. U této sítě se postupně snižuje latence, což je prodleva mezi dotazem a odpovědí sítě, která u sítí 2G dosahovala běžně kolem 1s.

„Sítě UMTS byly poprvé definovány standardizačním orgánem 3GPP v roce 1999. První UMTS standard se proto jmenuje UMTS Release 1999. Technické parametry, kterých sítě UMTS R99 však byly ve srovnání s EDGE katastrofální a bylo zjevné, že standard se prostě bude muset rychle posouvat dál. V roce 2001 byl proto schválen standard 3GPP Release 4 (zde už souvislost s rokem schválení vymizela), který zlepšoval přenosové parametry směrem od uživatele k síti (na tzv. uplinku). Ono 3GPP se přitom v názvu často zaměňuje za UMTS. Pokud se tak hovoří o UMTS Release 4 (R4), má se na mysli souhrn dokumentů zahrnutý v 3GPP Release 4. Síť UMTS R4 byla chvíli k vidění také u Eurotelu (nyní O2) v ČR. Nicméně svými parametry by dnes jen těžko někoho uspokojila“ [19].

„Zajímavé to začíná být až od Release 5, který přinesl technologii HSDPA (High Speed Downlink Packet Access). Ta výrazným způsobem zvýšila rychlost datových přenosů směrem k uživateli. Skok z původních 384 kbit/s na 1,8 a posléze 3,6 Mbit/s (myšleno na celou buňku) byl natolik markantní, že se toho telekomunikační marketéři ihned chytili

a začali síť UMTS R5 označovat jako 3,5G. Právě rychlost 3,6 Mbit/s. je přitom prozatím (na příštích pár týdnů) v ČR tou určující pro všechny tři operátory. Navíc softwarová úprava v řízení radiové části sítě UMTS, kterou HSDPA je, se začala označovat jako samostatná technologie. Nežřídká se tak dočtete u mobilního telefonu, že nabízí podporu pro síť UMTS a HSDPA“ [19].

„Release 6 pak přinesl další úpravy, z nichž nejpodstatnější bylo zrychlení datových přenosů směrem od uživatele do sítě shrnuté pod název HSUPA. I zde marketéři promptně zareagovali, a síť UMTS R6 začali hned označovat za 3,75G. Někdy se také používá označení HSPA, které značí, že daná síť nabízí HSUPA i HSDPA. Ohledně podpory pro Release 6 jsou někteří čeští mobilní operátoři poněkud skoupí na slovo, nicméně lze předpokládat, že v nějaké formě ji podporují taktéž všichni tři. U sítě UMTS R6 se pak počítá s rychlostí až 14,4 Mbit/s na downlinku (na buňku!) a až 5,76 Mbit/s na uplinku“ [19].

„Release 7 pak přichází s technologií HSPA+, kde ono „+“ označuje nárůst přenosových rychlostí na dvojnásobek oproti HSPA – tedy zpočátku na 28 Mbit/s na downlinku (na buňku) a 11,5 Mbit/s na uplinku, přičemž tento standard definuje i rychlosti až 56 Mbit/s a 22 Mbit/s, které však zatím nikdo nepodporuje ani na straně sítě ani na straně koncových zařízení. HSPA+ bývá označováno poměrně vtipně jako 3,9G“ [19].

„Zatím posledním schváleným standardem UMTS je Release 8. Ten přináší natolik významných změn, že se marketingová oddělení rozhodla už těmto sítím neříkat UMTS, nýbrž LTE (Long Term Evolution). LTE je také označováno jako mobilní síť 4. generace. Pro síť 4G přitom zatím panuje obecná shoda, že se přestane komunikovat teoretická a nikdy nedosažitelná rychlost na buňku, ale zákazníkům se začnou opět sdělovat rychlosti, kterých oni sami v praxi mohou dosáhnout“ [19].

Hlavní bezpečnostní mechanismy v 3G síti jsou založeny na autentizaci a šifrování. Síť 3G používá novou verzi karty SIM tzv. USIM. Používá zabezpečovací funkce f1, f1*, f2, f3, f4, f5, f5*, což jsou protokoly a algoritmy pro ochranu dat. Skupina těchto algoritmů se označuje názvem „MILENAGE“. Algoritmy f1 a f2 zabezpečují autentizaci. Algoritmy f2, f3, f4 se starají o generování klíčů. Algoritmy f1* a f5* zabezpečují resynchronizaci a jejich základem je algoritmus Rijndael. Algoritmy f8 a f9 se označují názvem „KASUMI“. Algoritmus f8 slouží k ochraně důvěrnosti a algoritmus f9 k ochraně integrity dat. Lepší zabezpečení systému UMTS je mimo jiné dáno předpokládaným využitím různých aplikací například v elektronickém obchodu, kde je nutné přenášet citlivé osobní údaje, kterými mohou být třeba čísla účtů a hesla. Bezpečnost v síti je rozdělena do dvou směrů:

- Bezpečnost přístupové sítě UTRAN (Rádiová přístupová síť)
- Bezpečnost na aplikační vrstvě

Na rozdíl od sítě GSM je autentizace v síti oboustranná. Autentizuje se jak mobilní stanice, tak síť mobilní stanici. Využívá se „generického protokolu EAP-AKA“ (Extensible Authentication Protocol - Authentication and Key Agreement). Tento protokol je složen právě z algoritmů MILENAGE.

3 KRYPTOGRAFICKÉ ALGORITMY V BEZDRÁTOVÝCH SÍTÍCH

3.1 Rijndael

Název tohoto algoritmu je odvozen od jmen jeho tvůrců Joan Daemen a Vincent Rijmen. Používá se u wifi sítí se zabezpečením WPA2, a v sítích 3G. Tento algoritmus splňuje požadavky AES a je považován za jeho standart. Byl navržen tak aby odolával všem známým útokům, byl implementačně jednoduchý a šel implementovat na všech platformách. Je to symetrický blokový algoritmus, používá tajný klíč o velikosti 128, 196 nebo 256 bitů, a délka bloku 128 bitů. Parametry AES pro šifrování jsou v tabulce 2.

Délka klíče(bit)	128	192	256
Velikost bloku(bit)	128	128	128
Počet cyklů	10	12	14
Délka klíče cyklu(bit)	128	128	128
Délka rozšířeného klíče(bit)	176	208	240

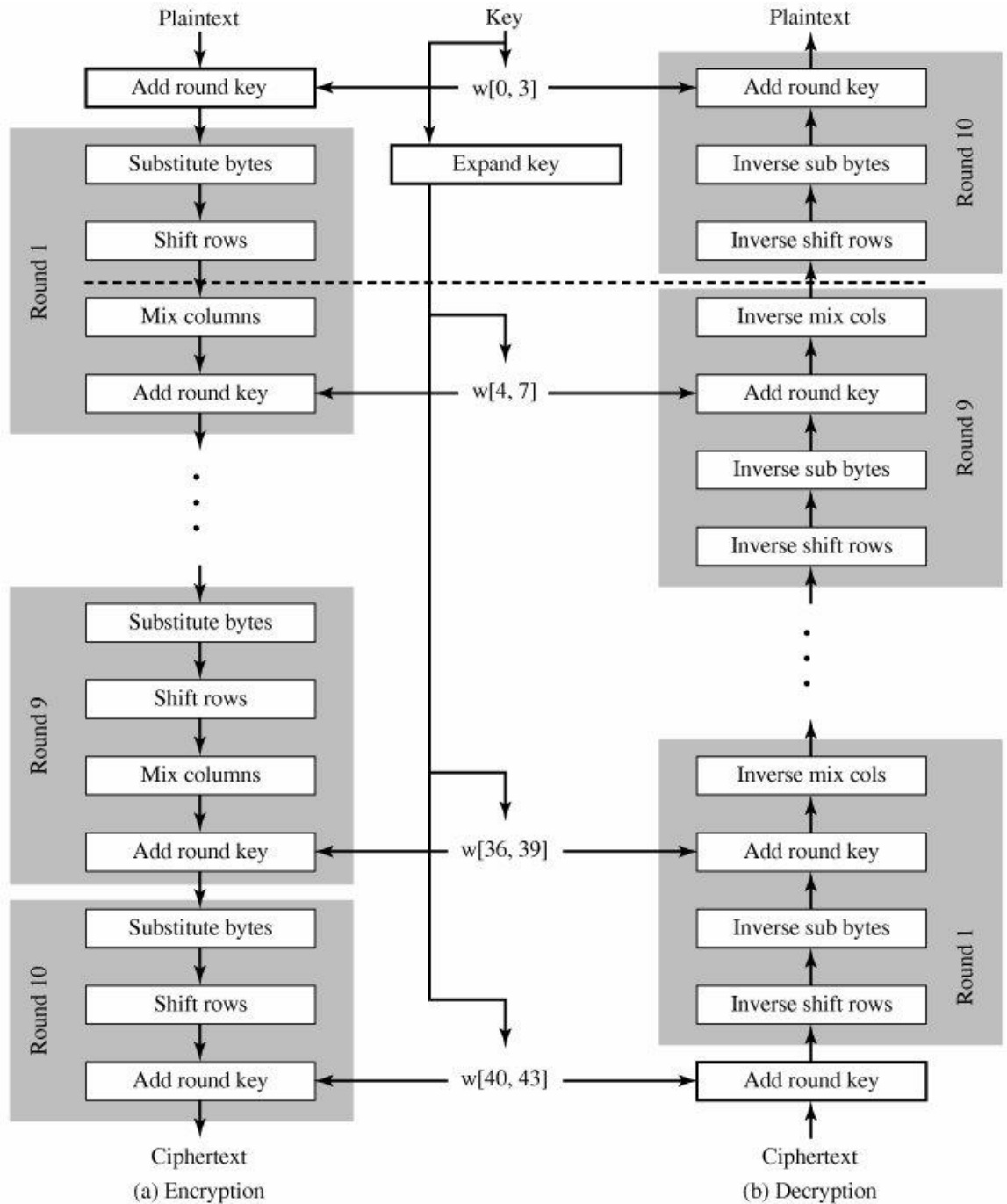
Tabulka 2: Parametry AES

V případě dat, která jsou delší, se data rozdělí na 128bitové bloky. V případě dat, která jsou kratší, se blok doplní na potřebnou délku 128 bitů. Princip šifrování je popsán v následujících krocích:

1. Vstupem do algoritmu je blok dat o délce 128 bitů a šifrovací klíč o délce 128 bitů. Vstupní data jsou uspořádána do tabulky 4*4. Každá buňka tabulky obsahuje jeden bajt. Stejným způsobem je do tabulky uspořádán šifrovací klíč. Výsledek je na obr. 9.

$a_{0,0}$	$a_{0,1}$	$a_{0,2}$	$a_{0,3}$	$k_{0,0}$	$k_{0,1}$	$k_{0,2}$	$k_{0,3}$
$a_{1,0}$	$a_{1,1}$	$a_{1,2}$	$a_{0,3}$	$k_{1,0}$	$k_{1,1}$	$k_{1,2}$	$k_{0,3}$
$a_{2,0}$	$a_{2,0}$	$a_{2,2}$	$a_{2,3}$	$k_{2,0}$	$k_{2,0}$	$k_{2,2}$	$k_{2,3}$
$a_{3,0}$	$a_{3,1}$	$a_{3,2}$	$a_{3,3}$	$k_{3,0}$	$k_{3,1}$	$k_{3,2}$	$k_{3,3}$

Obrázek 9: Uspořádání bloku šifrování



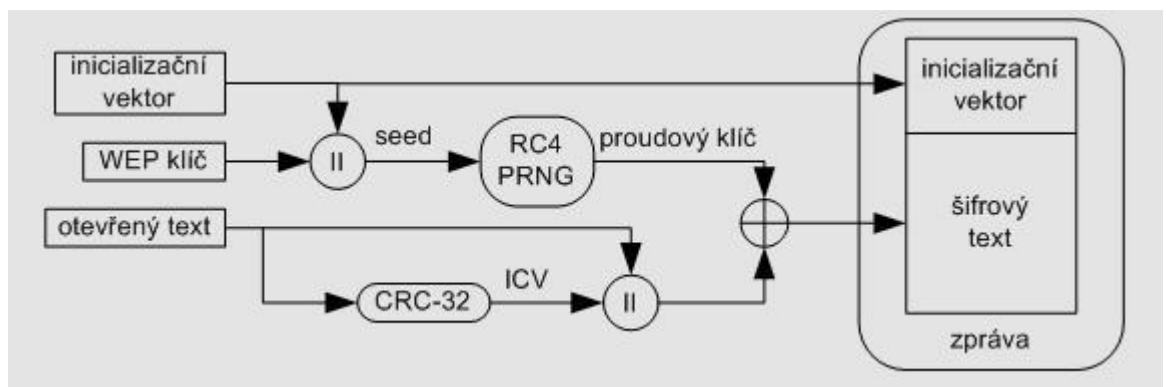
Obrázek 10: Princip šifrování Rijndael [18]

2. Z řetězce a hlavního klíče je odvozen klíč pro daný cyklus
3. Dojde k nahrazení jednotlivých bajtů v tabulce za použití tzv. S-boxu
4. Provede se záměna řádků
5. V každém sloupci se zkombinují všechny bajty
6. Vytvoří se podklíč pro další iteraci
7. Následuje další iterace, pokud nebyla poslední

3.2 RC4

Šifru RC4 navrhl Ron Rivest z RSA Security v roce 1987. Jedná se o proudovou šifru s tajným klíčem. Její výhodou je jednoduchost implementace a její rychlost. Není však zcela bezpečná. Používá se k šifrování dat ve wifi sítích s protokolem WEP a WPA.

„WEP používá nevhodně aplikovanou proudovou šifru RC4 (Rivest Cipher verze 4). Podrobný popis algoritmu RC4 (vyvinutý Ronem Rivestem v roce 1987) byl známý pouze osobám, které podepsali důvěrný dodatek, neboť byl ve vlastnictví RSA Data Security, Inc. V září roku 1994 však anonymní odesílatel uveřejnil zdrojový kód algoritmu, a tak se rychle rozšířil po celém světě. RC4 je ochranná známka RSA Data Security, Inc., takže je zveřejněný údajný RC4 algoritmus označován jako ARC4 (Alleged RC4). Již v roce 1997 ve své publikaci J. Golik poukázal na statistickou slabinu generátoru klíče u ARC4. V roce 2001 S. Fluhrer, I. Mantin a A. Shamir publikovali objevenou slabinu v algoritmu generování klíčů, zjistili, že existuje množina tzv. slabých klíčů, a na základě této myšlenky provedli pasivní útok na šifrovaný text. Algoritmus byl oficiálně označen jako zastaralý a nedoporučený bohužel až v roce 2004. Naneštěstí se i přesto na počátku roku 2009 vyskytují veřejně dostupné sítě zabezpečené pomocí WEP, jež je implementována pouze z důvodů zpětné kompatibility bezdrátových systémů“ [16].



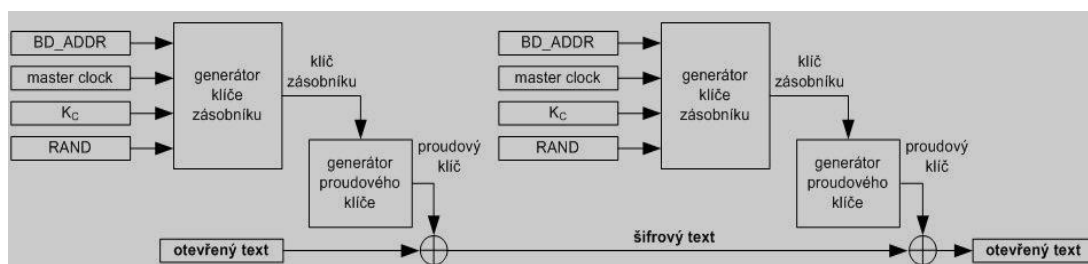
Obrázek 11: Zapouzdření WEP a začlenění RC4 [16]

Vstupem do algoritmu je klíč o délce až 256 bajtů. Nejvíce se používá 40 nebo 128 bitů.

3.3 E0

„Proudová šifra E0 byla zkonstruována pro šifrování datového toku bezdrátového komunikačního protokolu pro relativně krátké vzdálenosti (desítky metrů) zvaného Bluetooth (IEEE 802.15). E0 v inicializační fázi generuje klíč pro blok dat, v druhé fázi produkuje

jednotlivé bity proudu klíče na principu Massey-Rueppelova generátoru klíče. Ve třetí fázi se pak realizuje vlastní šifrování resp. dešifrování. I přesto, že byl Massey-Rueppelův algoritmus generování klíče navržen s ohledem na známé kryptoanalytické metody, je náchylný na korelační útoky. Pravděpodobnost úspěchu takového útoku snižuje vysoká resynchronizační frekvence (podle doporučení měla být vždy po odeslání bloku dat E0 resynchronizována). Každé zařízení Bluetooth má svou jedinečnou 48bitovou adresu BD_ADDR, která je použita k šifrování stejně jako šifrovací klíč K_C , 26 bity master clock a 128bitová náhodně vygenerovaná hodnota RAND“. [16] Princip šifrování je na obr. 11.



Obrázek 12: Princip šifry E0

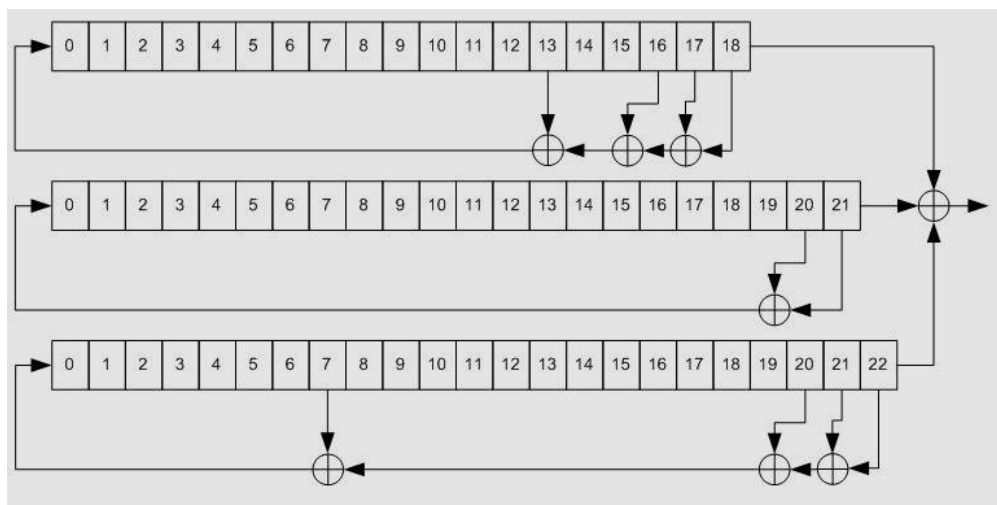
3.4 A5

A5 je proudová šifra vyvinutá k šifrování GSM hovoru mezi mobilní stanicí a základnovou stanicí BTS (Base Transceiver Station). Hovor v síti operátora, tj. od BTS přes BSC (Base Station Controller) až do ústředny MSC (Mobile Switching Center), není dále šifrován, takže ho lze odposlouchávat. Existuje v těchto variantách:

- A5/0 – neposkytuje žádné zabezpečení
- A5/1 – původní algoritmus používaný v Evropě, efektivní délka klíče je 54bitů, protože prvních deset bitů je nulových
- A5/2 – kryptograficky slabší varianta algoritmu používaná v USA
- A5/3(Kasumi) – silný šifrovací algoritmus 3GPP

„Šifra A5/1 byla vyvinuta již v roce 1987 v USA. Ačkoliv byl algoritmus A5/1 spolu s algoritmem její nástupkyně A5/2 utajován, unikly v roce 1994 informace o jejich obecné struktuře a v roce 1999 byly oba algoritmy zpětným inženýrstvím detailně zrekonstruovány. A5/1 využívá klíč délky 64 bitů spojený s veřejně známým číselným rámcem dlouhým 22 bitů. V implementaci GSM je ovšem deset bitů klíče pevně nastaveno na nulu, takže je efektivní délka klíče 54 bitů. To dělá z A5/1 šifru méně bezpečnou, než byl např. DES s klíčem délky 56 bitů. Teoretický útok pomocí předem vypočítaných stavů prezentoval v

roce 1997 J. Golić. Šifra A5/2 byla vyvinuta o dva roky později především pro oblast Asie a východní Evropy a byla podstatně slabší. V průběhu času byla nahrazována původní A5/1, neboť za velmi krátkou dobu po vydání byla možné šifru prolomit i na běžném osobním počítači [10]. Teprve od roku 2006 podle asociace GSMA (GSM Association) nepodporují mobilní telefony algoritmus A5/2, asociace 3GPP ji však ve svých standardech jako volitelnou stále podporuje. Až v roce 2006 předvedli E. Barkan, E. Biham a N. Keller útoky v reálném čase a možnost prolomit uloženou zachycenou komunikaci později“ [16]. Princip šifry A5/1 je znázorněn na obrázku 13.

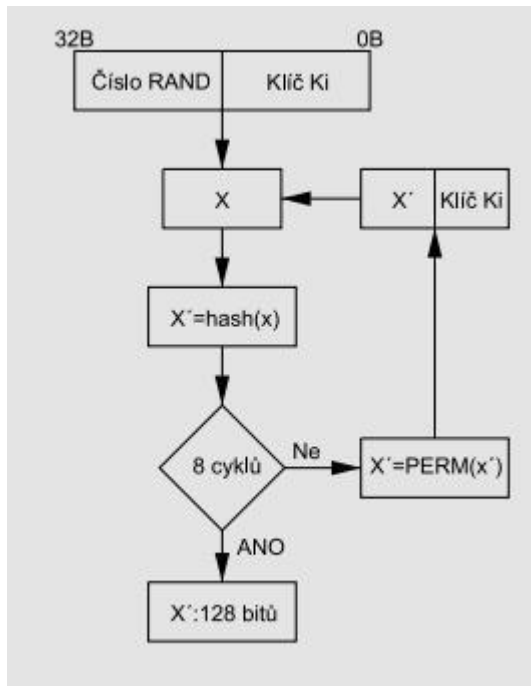


Obrázek 13: Princip šifry A5/1 [16]

3.5 COMP128

Pracuje nad 256bitů dlouhým vektorem ($X[0..31]$ of byte) v osmi cyklech. Při inicializaci algoritmu se horní polovina vektoru ($X[16]..X[32]$) naplní náhodným číslem $RAND(128b=16B)$ a spodní polovina se vždy před začátkem každého z těchto osmi cyklů přepíše stejně velkým klíčem K_i ($X[0]..X[15]$). Po provedení každého cyklu zůstává mezi-výsledek v horní polovině vektoru, zatímco spodní polovina se vždy znovu přepíše hodnotou K_i . A takto se postupuje do dalšího cyklu algoritmu. Každý z těchto osmi průběhů cyklu obsahuje další cyklus, jenž se pětkrát opakuje a provádí úpravu hodnot v jednotlivých bajtech vektoru X . Průběhům tohoto cyklu se říká rundy. V nich se vektor rozděluje na 16 dvojic bajtů, které se navzájem ovlivňují (viz program a hodnoty v tabulce). Podstatné přitom je, že tabulky ($table[0] .. table[4]$) zúží vždy výstup z dané rundy o jeden bit. Po provedení všech pěti rund je v jednotlivých bytech vektoru X pouze čtyřbitový obsah. S ním

se pak s výjimkou posledního z osmi průběhů cyklu provádí ještě bitová permutace (viz algoritmus) [20].



Obrázek 14: Princip algoritmu COMP128

3.6 CRC 32

Používá se ke kontrole dat, jestli při přenosu nedošlo k chybě. Mimo jiné je používána ve wifi sítích se zabezpečením WEP. Vyznačuje se tím, že z kódového slova vznikne cyklickým posuvem jiné slovo cyklického kódu. Při výpočtu CRC 32 se využívá logické operace XOR (obrázek 15). Spolehlivost této metody spočívá na volbě dělicího polynomu. Princip s osmibitovým CRC je znázorněn na obrázku 16.

A	B	A xor B
0	0	0
0	1	1
1	0	1
1	1	0

Obrázek 15: Pravdivostní tabulka XOR

1 1 1 0 1 0 1 0 0 0 0 0 0 0 0 0	
xor	
1 0 0 0 0 0 1 1 1	1. krok
=	
0 1 1 0 1 0 0 1 1 0 0 0 0 0 0 0	
xor	
1 0 0 0 0 0 1 1 1	2. krok
=	
0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 0	
xor	
1 0 0 0 0 0 1 1 1	3. krok
=	
0 0 0 0 1 0 0 0 1 0 1 0 0 0 0 0	
xor	
1 0 0 0 0 0 1 1 1	4. krok
=	
0 0 0 0 0 0 0 0 1 0 0 1 1 0 0 0	
- - - - - - - - 1 0 0 1 1 0 0 0	výsledek

Obrázek 16: Princip dělení CRC

3.7 Diffie-Hellmann

Kryptografický systém, který umožňuje vytvořit šifrované spojení přes nechráněný komunikační kanál bez předem dohodnutého klíče. Klíč se mezi komunikujícími stranami neposílá, ale obě komunikující strany si ho vytvoří samy. Nevýhodou je že tento způsob neumožňuje autentizaci účastníků. Tento systém je založen na problému diskrétního algoritmu.

Popis algoritmu:

Komunikace probíhá následujícím způsobem:

1. Účastníci se veřejně domluví na použitém modulu a základu.
2. Každý z účastníků si zvolí svůj exponent (nesoudělný s modulem).
3. Každý z účastníků umocní základ modulárně základ na svůj exponent a výsledek pošle dalšímu účastníkovi.
4. Algoritmus končí, když je každý z původních základů zpracován každým účastníkem.

4 NEDOSTATKY ZABEZPEČENÍ SÍTĚ GSM

Šifrování se v GSM provádí pouze mezi mobilní stanicí a BTS. Potom se nešifrovaná data přenáší sítí GSM až k BTS která komunikuje s druhou mobilní stanicí, kde opět dochází k šifrování dat. Toto umožňuje odposlech přenášených dat operátorem. GSM používá slabé šifrování, které lze prolomit. Navíc je autentizace v síti jednostranná, což umožňuje útok pomocí falešných základnových stanic. Dalším nedostatkem je klíč uložený v kartě SIM. Při zjištění tohoto klíče je možné SIM kartu naklonovat.

„V dubnu 1998 se skupině kryptologů složené z pánů Marca Bricena, Iana Goldberga a Davida Wagnera podařilo objevit metodu možnou zjistit klíč uložený v SIMkartě. Útok lze uskutečnit jen tehdy, máme-li k dispozici SIMkارتu. Karta se vloží do speciálního klonovacího zařízení spojeného s počítačem. Kartě jsou potom předkládány určité výzvy a analyzovány jsou reakce karty. Celkem je potřeba vznést cca 150 000 speciálně vybraných dotazů. Klonovací zařízení, které zkonstruovali, mohlo vyřídít 6.25 dotazů za vteřinu. K útoku s tímto zařízením je tedy potřeba asi 8 hodin. Pokud útočník vlastní toto zařízení získá vaši SIM kartu na tuto dobu, je schopen získat klíč v ní uložený a vyrobit klon vaší SIM karty. Může se potom do sítě autentizovat jako vaše SIM karta a účtovat hovorné na váš účet“ [21].

Z kapitoly vyplývá, že kryptografické možnosti ochrany dat sítě GSM jsou dnes už velice zastaralé. Není problém odposlouchávat hovory, sms případně přenášená data pomocí sítě GSM, ať už přímo operátorem, nebo třetí osobou vlastníci příslušné zařízení. Dalšími možnostmi, jak se odposlechům bránit rozebírá následující kapitola.

4.1 Šifrované telefony

Odposlech mobilní komunikace přes síť GSM je stále snadnější. Jenom v české republice je sedm subjektů, které mohou takovýto odposlech nařídit. Další rizika sebou nesou zařízení, pomocí kterých lze tuto komunikaci odposlouchávat ilegálně. Na trhu je několik firem, které šifrované telefony nabízejí, ať už se jedná přímo o hardwarové šifrovací telefony, nebo o programy které tuto funkčnost nabízejí.

Šifrované telefony znemožňují komukoliv odposlech Vaší vzájemné komunikace, kdy se zašifruje mluvené slovo, zašifrují se SMS, zašifrují se obrázky a také se zašifruje rychlý CHAT mezi účastníky spojení. Typy šifrovaných aplikací:

- šifrované mobilní telefony

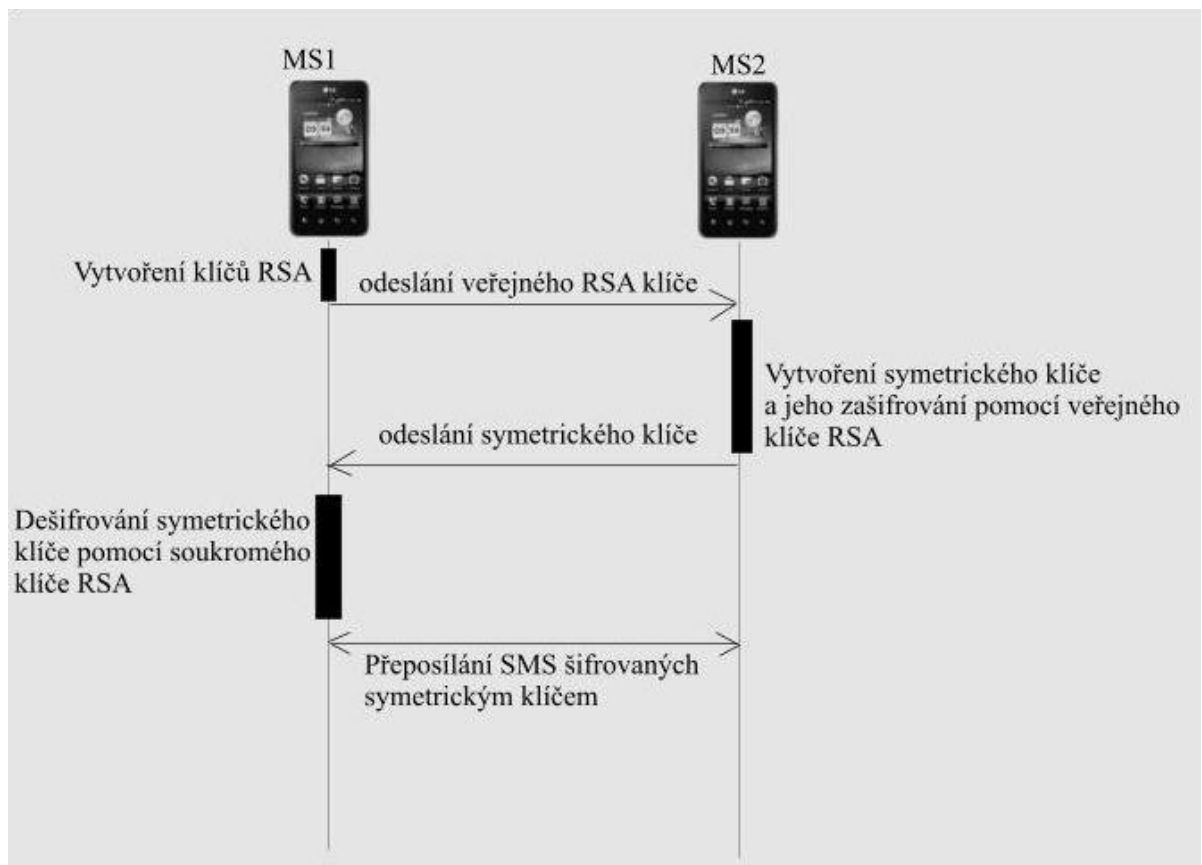
- šifrované pevné linky
- šifrované SMS – krátké textové zprávy
- šifrované komunikace typu CHAT
- šifrované přenosy JPEG – obrázky
- šifrované IP telefony, faxy ...
- šifrované e-maily pro bezpečnou komunikaci mimo telefonní spojení

II. PRAKTICKÁ ČÁST

5 APLIKACE PRO ŠIFROVÁNÍ SMS NA PLATFORMĚ ANDROID

5.1 Specifikace

Aplikace je praktickou ukázkou, jak lze řešit kryptografické zabezpečení odesílaných SMS v mobilním telefonu pro operační systém Android. Je naprogramovaná v jazyce Java ve vývojovém prostředí Eclipse. K odladění programu byl použit emulátor a aplikace byla otestovaná na telefonu Sony Ericsson Xperia Active s operačním systémem Android 4.0.4. V programu je využito kombinace symetrické a asymetrické kryptografie. K šifrování SMS zpráv je využito algoritmu Rijndael, který je dnes považován za standart AES. Tento algoritmus je popsán v kapitole 3.1. K výměně klíče algoritmu AES slouží RSA kryptografie.



Obrázek 17: Sekvenční diagram principu šifrování zpráv

5.2 RSA kryptografie

Algoritmus pracuje v těchto krocích:

1. Určí se dvě velká prvočísla p a q

2. Spočítá se: $n = p * q$
3. Spočítá se: $m = (p-1) * (q-1)$
4. Určí se prvočíslo K_1
5. Vypočítá se: $K_2 = (1 + mr)/K_1$ kde $r = \left((m^{k_1-2} \bmod K_1) (K_1 - 1) \right) \bmod K_1$
6. Šifrování zprávy se provede jako $S' = S^{k_1} \bmod n$
7. Dešifrování zprávy se provede jako $S = S'^{k_2} \bmod n$ [22]

Pro RSA kryptografii platí, že šifrovaná zpráva musí mít kratší délku než číslo n . Bezpečnost této metody je dána délkou klíče. K šifrování zprávy je tedy zapotřebí K_1 a číslo n . Nebezpečí prolomení RSA spočívá ve faktorizaci čísla n .

„Asymetrická kryptografie vychází z principu, že zpráva je kódována jedním klíčem a dekodována jiným. Klíče jsou matematicky závislé, ale není jednoduché ze znalosti jednoho klíče spočítat druhý. Kryptografická metoda RSA se opírá o obtížnost faktorizace čísla. Práce s šifrou probíhají v logaritmickém čase, zatímco rozlomení šifry vyžaduje v zásadě čas lineární“ [22].

Obecně jsou za kvalitní považovány asymetrické šifry s délkou klíče nad 700 bitů. RSA většinou využívá klíče délky 517 – 4096 bitů [23].

V programu je použit klíč o délce 788 bitů. Je to z toho důvodu, že pomocí jedné sms lze odeslat pouze určité množství dat. Tento nedostatek by se dal řešit přeposláním klíčů ve více než jedné SMS, nicméně i tak je rozluštění šifry velmi zdlouhavé.

Otestování obtížnosti rozlomení RSA šifry hrubou silou bylo provedeno za pomoci jazyka Scheme [22]. Zdrojový kód je uveden v příloze. Obrázek 18 ukazuje, jak roste čas potřebný k rozlomení šifry vzhledem k délce klíče. Délka klíčů je v rozmezí 24 až 51 bitů. Čas v prostředním sloupci je reálný čas potřebný k rozlomení šifry udaný v ms. Z obrázku je patrné, že rozluštění 788 bitového klíče touto metodou je prakticky nemožné.

5.3 Uživatelské rozhraní programu

Program se skládá ze tří aktivit. První úvodní aktivita slouží k zadání uživatelského hesla. Tato funkčnost není v programu implementovaná a slouží pouze jako ukázka zabezpečení zneužití programu třetí osobou. Další aktivita by měla sloužit k nastavení tohoto hesla.

Hlavní aktivita programu je na obrázku 19. Funkce programu je následující:

```

Welcome to DrScheme, version 209.
Language: Pretty Big (includes MrEd and Advanced).
Teachpack: C:\Users\Zdenek1\Documents\skola\bc\Bc\1semestr\Paradigmata prog
cpu time: 0 real time: 0 gc time: 0      24 bit RSA
"zp"
cpu time: 16 real time: 16 gc time: 0    27 bit RSA
"zp"
cpu time: 16 real time: 15 gc time: 0    30 bit RSA
"zp"
cpu time: 78 real time: 72 gc time: 0    33 bit RSA
"zp"
cpu time: 843 real time: 1006 gc time: 296 36 bit RSA
"zp"
cpu time: 1201 real time: 1225 gc time: 281 39 bit RSA
"zp"
cpu time: 9609 real time: 10934 gc time: 2965 42 bit RSA
"zp"
cpu time: 23463 real time: 26265 gc time: 7301 45 bit RSA
"zp"
cpu time: -405036 real time: 26610 gc time: 7523 48 bit RSA
"zp"
cpu time: 129184 real time: 141921 gc time: 39382 51 bit RSA
"zp"
>
    
```

Obrázek 18: doba potřebná k rozlomení RSA



Obrázek 19: Hlavní okno programu

1. Po stisku tlačítka „Pick contact“ se otevře telefonní seznam, a vybere se příjemce zprávy. Telefonní číslo je rovněž možné po stisknutí textboxu napsat pomocí softwarové klávesnice
2. Po vybrání příjemce je nutné stisknout tlačítko „Send Public Key“. Toto tlačítko odešle veřejný klíč příjemci. Tato událost je indikovaná pomocí tzv. Toastu, což je zpráva, která se zobrazuje na displeji. V případě že jsme příjemci veřejného klíče, je třeba zkopírovat text sms do textboxu pro sms pomocí tlačítka „Open sms“, a stisknout tlačítko „Decrypt sms“. V tomto případě se odešle symetrický klíč zašifrovaný pomocí veřejného klíče RSA.
3. Po doručení veřejného klíče je třeba klíč načíst do textboxu pro zprávy sms a stisknout tlačítko „Decrypt sms“.
4. Po zobrazení Toastu s textem „write sms“ je možné pomocí tlačítka „Send sms“ přeposílat sms zašifrované symetrickým klíčem.

Sekvenční diagram tohoto postupu je na obrázku 17.

5.4 Technická dokumentace

5.4.1 Popis tříd

5.4.1.1 *MainActivity*

Třída implementuje obsluhu událostí všech ostatních komponent využitých v programu. Je asociovaná se všemi ostatními třídami. Je v ní implementovaná logika přeposílání klíčů.

5.4.1.2 *SMSSender*

Je zapouzdřením třídy SMSManager, která slouží pro odesílání sms.

5.4.1.3 *TagReader*

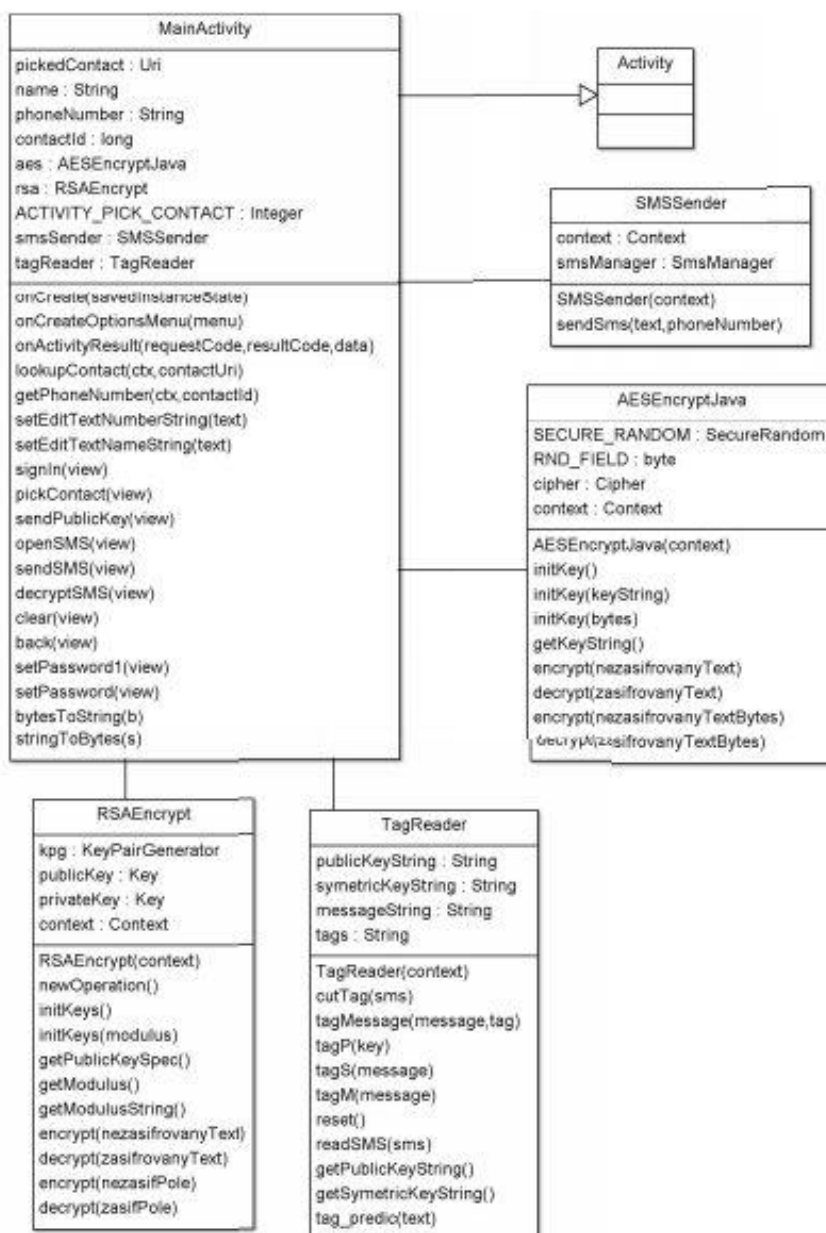
Sms které jsou odesílané a přijaté programem začínají tagem. Třída se stará o správné přečtení tagu, a jeho odstranění ze zprávy a přidání do nové zprávy.

5.4.1.4 RSAEncrypt

Třída je zodpovědná za RSA šifrování. Vytváří veřejný a soukromý klíč, obsahuje funkce pro šifrování pole bajtů a šifrování textu. Je využita pro zašifrování a odšifrování symetrického klíče. K šifrování využívá třídy z knihovny „java.security“ a „javax.crypto“.

5.4.1.5 AESEncryptJava

Třída implementuje symetrickou šifru AES. Využívá knihovny „java.security“ a „javax.crypto“. Slouží k šifrování sms zpráv.



Obrázek 20: Diagram tříd

ZÁVĚR

Cílem této práce bylo ukázat na nedostatky v zabezpečení bezdrátových sítí. Teoretická část se zabývá historií šifrování, podává přehled historických šifrovacích metod až po šifrovací metody používané v dnešní době. Další část poskytuje ucelený přehled bezdrátových sítí a v jednotlivých sítích popisuje způsoby, jakými jsou řešeny problémy spojené s autentizací a šifrováním přenášených dat. Práce popisuje způsoby zabezpečení různých druhů bezdrátových sítí. Tyto sítě lze rozdělit na dva hlavní typy. Prvním typem jsou sítě s malým dosahem, což jsou technologie určené pro budování lokálních sítí sloužících pro připojení několika zařízení. Jsou to například technologie Bluetooth, ZigBee nebo IrDa. Druhou skupinou jsou sítě s větším dosahem, určené primárně pro připojení lokálních sítí k internetu nebo propojení budov. Dalšími příklady bezdrátových sítí může být systém GPS nebo nejpoužívanější bezdrátová technologie GSM. U každého typu sítě jsou uvedeny způsoby jejich kryptografického zabezpečení, z čehož vyplývají nedostatky v jejich zabezpečení. Praktická část se zaměřuje na nedostatky v zabezpečení sítě GSM. Tento systém je již poměrně zastaralý a má několik bezpečnostních rizik. Největším rizikem je, že data v síti GSM se šifrují pouze mezi mobilním telefonem a stanicí BTS. V síti operátora jsou tyto data přenášena v nešifrované podobě. Na trhu je několik firem zabývajících se touto problematikou. Tyto firmy nabízejí už šifrované telefony, nebo programy nabízející šifrované spojení pomocí datových služeb. Jejich služby jsou však velmi drahé. Formou projektu je vypracované řešení zabezpečení přenosu sms v síti GSM. Tímto řešením je šifrovací program, který je vytvořen v jazyce Java pro mobilní telefony s operačním systémem Android. Program využívá kombinace RSA kryptografie a šifrovací metody AES. RSA kryptografie je považována za bezpečnou, pokud je použita vhodná délka šifrovacího klíče. Metoda AES je dnes standardem v symetrické kryptografii. U tohoto programu bylo přihlédnuto k tomu, že pomocí sms lze přenést pouze určité množství dat. Symetrické šifrovací metody jsou všeobecně považované za bezpečné, pokud používají klíč o minimální délce 70 bitů, což je pro tyto účely plně dostačující. Asymetrická kryptografie se považuje za bezpečnou, pokud je délka klíče větší než 700 bitů. V tomto programu je použit šifrovací klíč o délce 788 bitů, což je maximum pro přenesení v jedné sms. Pomocí programu pro rozlomení RSA kryptografie v jazyce Scheme je nastíněna obtížnost rozlomení RSA kryptografie hrubou silou. Nutno podotknout že existují lepší algoritmy, ale ani ty pro tuto délku klíče nejsou velkou hrozbou. Další možností by bylo přeposlat veřejný klíč ve více než jedné sms.

CONCLUSION

The aim of this thesis was to show the flaws in the security of wireless networks. The theoretical part deals with the history of encryption, provides an overview of historical encryption methods to the encryption method used today. Another section gives an overview of wireless networks and individual network describes the ways in which they are dealt with problems related to authentication and encryption of transmitted data. This work describes methods for securing different types of wireless networks. Such a computer network can be divided into two main types. The first type of networks are short range, which are technologies designed to build local networks for connecting multiple devices. Such as Bluetooth, ZigBee or IrDA. The second group is a network with greater range designed primarily for connecting local networks to the Internet, or link building. Other examples of wireless networks can be a GPS system or the most widely used GSM wireless technology. For each type of network are the ways of cryptographic security, leading to gaps in security. The practical part focuses on security weaknesses SITS GSM. This system is already quite outdated and has several security risks. The biggest risk is that the data in the GSM network are encrypted only between the mobile phone and the BTS station. The network operator's data are transmitted in unencrypted form. On the market there are several companies dealing with this issue. These companies offer either encrypted phones, or programs offering encrypted connection using data services. Their services are very expensive. The project format is developed security solutions for the SMS in GSM network. This solution is an encryption program that is created in Java for mobile phones running Android. The program uses a combination of RSA cryptography and AES encryption methods. RSA cryptography is considered safe when used appropriately length of the encryption key. AES method is now standard in symmetric cryptography. In this program, consideration was given to the fact that SMS can only transfer a certain amount of data. Symmetric encryption methods are generally considered safe when used with a minimum key length of 70 bits, which for these purposes is sufficient. Asymmetric cryptography is considered safe if the key length is greater than 700 bits. The program uses an encryption key length of 788 bits, which is the maximum for the transfer in a single SMS. Use the program to break RSA cryptography in the language Scheme outlines the difficulty of breaking RSA cryptography brute force. It should be noted that there are better algorithms, but not those for the key length, is not a major threat. Another option would be to transfer the public key in more than one sms.

SEZNAM POUŽITÉ LITERATURY

- [1] <http://www.fi.muni.cz/usr/jkucera/pv109/2003/xbitto.htm>
- [2] TRÍSKA, David. Kryptografická ochrana, 2009
- [3] <http://www.krypta.cz/articles.php?ID=124>
- [4] <http://prf-czv.osu.cz/nabidka/seminar/data/Kryptografie.pdf> (Doc. Ing. Cyril Klimeš, CSc. Kryptografie.pdf)
- [5] VAUDENAY, Serge. A CLASSICAL INTRODUCTION TO CRYPTOGRAPHY Applications for Communications Security
- [6] http://kryptologie.uhk.cz/idea_cz.htm
- [7] COURTOIS, Nicolas T. Security Evaluation of GOST 28147-89 In View Of International Standardisation
- [8] www.bluetooth.com
- [9] <http://extranotebook.cnews.cz/technologie/bluetooth-bezpecnejsi-rychlejsi>
- [10] <http://www.palowireless.com>
- [11] http://www.jirkasvoboda.com/publikace/publikace_3.pdf
- [12] <http://www.ics.muni.cz/bulletin/articles/624.html>
- [13] Guillaume Lehembre. Bezpečnost Wi-Fi – WEP, WPA a WPA2
- [14] http://data.idnes.cz/soubory/tec_checktech/a041123_jma_skr_bezdratove-mobilni-komunikace.pdf
- [15] PETERKA, Jiří. K čemu je GPRS/ <http://www.earchiv.cz/b01/b0100001.php3>
- [16] <http://access.feld.cvut.cz/view.php?cisloclanku=2009080001>
- [17] <http://dotnet-snippets.com/dns/rc4-encryption-SID577.aspx>
- [18] STALLINS, William. Cryptography and Network Security Principles and Practices, Fourth Edition. 2005
- [19] <http://www.businessvize.cz/datove-prenosy-a-site/jak-se-vyznat-v-mobilnich-datovych-sitich-umts-hsdpa-hsupa-hspa-lte>
- [20] http://mobil.idnes.cz/jak-se-klonuji-sim-karty-0wt-/mob_tech.aspx?c=A020605_5070850_mob_tech

[21] <http://www.scard.org/gsm/gsm-faq.html>

[22] SKOUPIL, David. Programování v jazyce scheme 2./

<http://phoenix.inf.upol.cz/esf/ucebni/Programy%20a%20projekty%20v%20jazyku%20Scheme%20II.pdf>

[23] <http://kryptologie.uhk.cz/62.htm>

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

3G	3 generace
A0	proudová šifra
ACL	Asynchronous Connection-oriented logical transport
AES	Advanced Encryption Standard
AMPS	Advanced Mobile Phone System
ASCII	American Standard Code for Information Interchange
BTS	Base transceiver station
CRC32	Cyclic redundancy check
DSA	Digital Signature Algorithm
E5	proudová šifra
EAP-AKA	Extensible Authentication Protocol
EDGE	Enhanced Data rates for GSM Evolution
ESN	Electronic serial number
ETSI	European Telecommunications Standards Institute
FFD	Full functional device
GOST	Euroasijská rada pro normalizaci, metrologii a certifikaci
GPRS	General Packet Radio Service
GPS	Global Positioning System
GSM	Globální System pro Mobilní komunikaci
GTK	Group Transient Key
HLR	Home Location Register
HSCSD	High-Speed Circuit-Switched Data
HSDPA	High-Speed Downlink Packet Access
HSUPA	High-Speed Uplink Packet Access
IDEA	International Data Encryption Algorithm
IrDa	Infrared Data Association
L2CAP	Logical link control and adaptation protocol
LMP	Link management protocol
LTE	Long Term Evolution
MAC	Media Access Control
MS	Mobil station
NMT	Nordic Mobile Telephone
NSA	National Security Agency
OFDM	Orthogonal Frequency Division Multiplexing
PAL	Protocol adaptation layer
PBKDF	Password-Based Key Derivation Function
PIN	personal identification number
PMK	Pairwise Master Key
PTK	Pairwise Transit Key
RAM	random-access memory
RC4	ARCFOUR -šifrovací algoritmus
RFCOMM	Radio frequency communication
RFD	Reduced functional device
ROM	Read-Only Memory

RSA	Rivest, Shamir, Adleman
SDP	Session Description Protocol
SIG	Special Interest Group
SIM	subscriber identity module
SMS	Short message service
SRES	Signed Response
SSID	Service Set Identifier
SSL	Secure Sockets Layer
TACS	Total Access Communication System
TCP/IP	Transmission Control Protocol /Internet Protocol
TDMA	Time division multiple access
TCH	Traffic Channel
TKIP	Temporal Key Integrity Protocol
TMSI	Temporary Mobile Subscriber Identity
UMTS	Universal Mobile Telecommunications System
VLR	Visitor Location Register
WEP	Wired Equivalent Privacy
WPA	Wi-Fi Protected Access

SEZNAM OBRÁZKŮ

Obrázek 1: Rozdělení šifer	16
Obrázek 2 : Princip symetrického šifrování	16
Obrázek 3: Princip asymetrické kryptografie	19
Obrázek 4: Kombinace symetrické a asymetrické kryptografie	21
Obrázek 5: Zapouzdření TKIP [14]	35
Obrázek 6: Zapouzdření CCMP [14]	35
Obrázek 7 : schéma GSM sítě.....	38
Obrázek 8 : Generace tripletu [14]	41
Obrázek 9: Uspořádání bloku šifrování	48
Obrázek 10: Princip šifrování Rijndael [18].....	49
Obrázek 11: Zapouzdření WEP a začlenění RC4 [16]	50
Obrázek 12: Princip šifry E0	51
Obrázek 13: Princip šifry A5/1 [16]	52
Obrázek 14: Princip algoritmu COMP128	53
Obrázek 15: Pravdivostní tabulka XOR	53
Obrázek 16: Princip dělení CRC	54
Obrázek 17: Sekvenční diagram principu šifrování zpráv.....	58
Obrázek 18: doba potřebná k rozlomení RSA	60
Obrázek 19: Hlavní okno programu	60
Obrázek 20: Diagram tříd	62

SEZNAM TABULEK

Tabulka 1: Třídy složitostí	15
Tabulka 2: Parametry AES	48

PŘÍLOHA A: ZDROJOVÝ KÓD RSA

```

; b na n-tou s lineární složitostí
(define (expt b n)
  (if (= n 0) 1
      (* b (expt b (- n 1)))))
;-----
;pokud je n liché vrací true
(define (odd? n)
  (= (remainder n 2) 1))
;-----
; b na n-tou s logaritmickou složitostí
(define (expres-expt b n)
  (cond ((= n 0) 1)
        ((odd? n) (* b (expres-expt b (- n 1))))
        ((= n 2) (sqr b))
        (else (sqr (expres-expt b (/ n 2))))))
;-----
(define (expmod2 b n m)
  (remainder (expres-expt b n) m))
;-----
(define (expmod b n m)
  (cond ((= n 0) (remainder 1 m))
        ((= n 1) (remainder b m))
        ((odd? n) (remainder (* b (expmod b (- n 1) m) m))
        ((= n 2) (remainder (sqr (remainder b m) m))
        (else (remainder (sqr (expmod b (/ n 2) m) m))))))
;-----
(define (fermat-test n)
  (define rnd (xrandom n))
  (= rnd (expmod rnd n n)))
;-----
; x-krát zopakuje fermat test
(define (fermat-test-i n i last)
  (if (> i 0)
      (fermat-test-i n (- i 1) (and (fermat-test n) last))
      (and (fermat-test n) last)))
;-----
(define (expres-prvocislo? n)
  (fermat-test-i n 100 #t))
;-----
(define (generuj-cifry n)
  (if (= n 0) 0
      (+ (* (random 10) (expres-expt 10 (- n 1)))
          (generuj-cifry (- n 1)))))
;-----
(define (pocet-cifer n)
  (if (> (quotient n 10) 0)
      (+ 1 (pocet-cifer (quotient n 10)))
      1))
;-----
(define (xrandom n)
  (if (<= n 2147483647) (random n)
      (generuj-cifry (random (pocet-cifer n)))))
;-----
(define (generuj-liche n)
  (define m (generuj-cifry n))
  (if (odd? m)
      m
      (- m 1)))
;-----
(define (generuj-prvocislo-help p)
  (if (expres-prvocislo? p)
      p
      (generuj-prvocislo-help (- p 2))))

```

```

(define (generuj-prvocislo n)
  (generuj-prvocislo-help (generuj-liche n)))
;-----
;RSA
;-----
(define (vypocti-n p q)
  (* p q))
;-----
(define (vypocti-m p q)
  (* (- p 1) (- q 1)))
;-----
(define (vypocti-r k1 m)
  (remainder (* (expmod m (- k1 2) k1) (- k1 1)) k1))
;-----
(define (vypocti-r k1 m)
  (remainder (* (expmod m (- k1 2) k1) (- k1 1)) k1))
;-----
(define (vypocti-k2-help k1 m r)
  (/ (+ 1 (* m r)) k1))

(define (vypocti-k2 k1 p q)
  (vypocti-k2-help k1 (vypocti-m p q)
    (vypocti-r k1 (vypocti-m p q))))

;-----
(define (kodujiRSA s k1 n)
  (expmod s k1 n))
;-----
(define (dekodujRSA kod-s k2 n)
  (expmod kod-s k2 n))
;-----
;prevod znaku do ASCII a opačně
;-----
(define (text->cislo s)
  (if (= (string-length s) 0) 0
    (+ (char->integer (string-ref s 0))
      (* 1000 (text->cislo (substring s 1 (string-length s)))))))

(define (cislo->text n)
  (define (safe-integer->char n)
    (if (and (> n 0) (< n 256))
      (integer->char (floor n))
      (error "Chybna zprava")))
  (if (< n 1000)
    (string (safe-integer->char n))
    (string-append (string (safe-integer->char (remainder n 1000)))
      (cislo->text (quotient n 1000)))))
;-----
(define (kodujiRSA-s s k1 n)
  (expmod (text->cislo s) k1 n))
;-----
(define (dekodujRSA-s kod-s k2 n)
  (cislo->text (expmod kod-s k2 n)))
;-----
;celociselna odmocnina
;-----
(define (deli? test n)
  (= (remainder n test) 0))

;-----
(define (hledej-delitele test n)
  (cond ((>= test n) 0)
    ((deli? test n) test)
    (else (hledej-delitele (+ test 1) n))))
;-----
(define (urci-p n)
  (hledej-delitele 2 n))

```

```
(define (urci-q n p)
  (/ n p))

(define (rozlom-k2 k1 n)
  (define p (urci-p n))
  (vypocti-k2 k1 p (urci-q n p)))

(define (rozlomRSA zpr k1 n)
  (cislo->text (dekodujRSA zpr (rozlom-k2 k1 n) n)))
;-----
;(time (rozlomRSA 13859684 65537 58195979)) ;24 bit RSA
;(time (rozlomRSA 24400173 65537 260450243)) ;27 bit RSA
;(time (rozlomRSA 1416895870 65537 1806146191)) ;30 bit RSA
;(time (rozlomRSA 465444531 65537 20047444981)) ;33 bit RSA
;(time (rozlomRSA 53015240463 65537 427960758451)) ;36 bit RSA
;(time (rozlomRSA 1470836120705 65537 6783881244383)) ;39 bit RSA
;(time (rozlomRSA 11067970060359 65537 56038699006819)) ;42 bit RSA
;(time (rozlomRSA 84086654164697 65537 356103891252481)) ;55 bit RSA
;(time (rozlomRSA 103960205608798 65537 1039886083969331)) ;48 bit RSA
;(time (rozlomRSA 95321673566992673 65537 99291252833621743)) ;51 bit RSA

;(time (kodujRSA-s "zp" 65537 1039886083969331))

;(time (rozlomRSA 17417777124838706009514 251226863267 221823717366419704863563))
;(define p (generuj-prvocislo 8))
;(define q (generuj-prvocislo 8))
;(define n (vypocti-n p q))
;(define k1 65537)
;(define k2 (vypocti-k2 k1 p q))
;(define s "zp")
;(define kod-s (kodujRSA-s s k1 n))
;(define dekod-s (dekodujRSA-s kod-s k2 n))
;kod-s
;dekod-s
;k1;n
```

PŘÍLOHA B: ZDROJOVÝ KÓD SMS_ENCRYPTOR

//třída MainActivity

```
package com.example.smsencryptor;
import java.math.BigInteger;
import java.util.Arrays;
import android.annotation.TargetApi;
import android.app.Activity;
import android.content.Context;
import android.content.Intent;
import android.database.Cursor;
import android.net.Uri;
import android.os.Build;
import android.os.Bundle;
import android.provider.ContactsContract;
import android.view.Menu;
import android.view.View;
import android.widget.EditText;
import android.widget.Toast;

@TargetApi(Build.VERSION_CODES.GINGERBREAD)
public class MainActivity extends Activity {

    private Uri pickedContact;
    private String name;
    private String phoneNumber;
    private long contactId;
    private AESEncryptJava aes;
    private RSAEncrypt rsa;
    private static final int ACTIVITY_PICK_CONTACT = 42;
    private SMSSender smsSender;
    private TagReader tagReader;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);
        this.aes = new AESEncryptJava(this);
        this.rsa = new RSAEncrypt(this);
        this.tagReader = new TagReader(this);
        this.smsSender = new SMSSender(this);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is
present.
        getMenuInflater().inflate(R.menu.activity_main, menu);
        return true;
    }

    //do Uri pickedContact uloží uri na vybraný kontakt
    @Override
    protected void onActivityResult(int requestCode, int resultCode, Intent data) {
        super.onActivityResult(requestCode, resultCode, data);
        switch (requestCode) {
```

```

        case (ACTIVITY_PICK_CONTACT) :
            if (resultCode == Activity.RESULT_OK) {
                try{
                    pickedContact = data.getData();
                    this.name = this.lookupContact(this,
this.pickedContact);
                    this.phoneNumber = this.getPhoneNumber(this, con-
tactId);
                }
                catch(Exception e)
                {
                    this.setEditTextSMSString(e.getMessage());
                }

                this.setEditTextNumberString(this.phoneNumber);
                this.setEditTextNameString(this.name);
            }
        }
        break;
    }
}

private String lookupContact(Context ctx, Uri contactUri) {
    String name = "nepriradil se";
    String[] projection = new String[]{
        ContactsContract.Contacts._ID,
        ContactsContract.Contacts.DISPLAY_NAME_PRIMARY,
    };

    Cursor c = null;

    try{
        c = ctx.getContentResolver().query(contactUri, projection,
null, null, null);
    }
    catch(Exception e)
    {
        name = e.getMessage();
    }

    if (c != null && c.moveToFirst()) {
        this.contactId = c.getLong(0);
        name = c.getString(1);
        this.name = name;
    }
    if (c != null) {
        c.close();
    }

    return name;
}

private String getPhoneNumber(Context ctx, long contactId) {
    Cursor cursor =
ctx.getContentResolver().query(ContactsContract.CommonDataKinds.Phone.CONTENT_U
RI,
        new String[]{
            ContactsContract.CommonDataKinds.Phone.NUMBER},

```

```
ContactsContract.CommonDataKinds.Phone.CONTACT_ID + "= ?" +
        " AND " + ContactsContract.CommonDataKinds.Phone.TYPE +
        "=" + ContactsContract.CommonDataKinds.Phone.TYPE_MOBILE,
        new String[]{String.valueOf(contactId)},
        null);

    try {
        if (cursor.moveToFirst()) {
            return cursor.getString(0);
        } else {
            return null;
        }
    } finally {
        cursor.close();
    }
}

public String decryptAES(String text) {
    String result = null;
    try {
        result = aes.decrypt(text);
    } catch (Exception e) {
        // TODO Auto-generated catch block
        this.setTextSMSString("decryptAES chyba");
    }
    return result;
}

private void setEditTextNumberString(String text) {
    EditText et = ((android.widget.EditText) findViewById(R.id.editTextNumber));
    et.setText(text);
}

private void setEditTextNameString(String text) {
    EditText et = ((android.widget.EditText) findViewById(R.id.editTextName));
    et.setText(text);
}

private void setEditTextSMSString(String text) {
    EditText et = ((android.widget.EditText) findViewById(R.id.editTextSMS));
    et.setText(text);
}

private String getEditTextNumberString() {
    EditText et = ((android.widget.EditText) findViewById(R.id.editTextNumber));
    return et.getText().toString();
}

private String getEditTextSMSString() {
    EditText et = ((android.widget.EditText) findViewById(R.id.editTextSMS));
    return et.getText().toString();
}

public void signIn(View view) {
```

```

        setContentView(R.layout.activity_main);
    }

//-----
//handlery událostí
//-----

    public void pickContact(View view) {
        Intent intent = new Intent(Intent.ACTION_PICK, ContactsContract.Contacts.CONTENT_URI);
        startActivityForResult(intent, ACTIVITY_PICK_CONTACT);
    }

    public void sendPublicKey(View view) {
        String number = this.getTextNumberString();
        rsa.initKeys();
        String mod = rsa.getModulusString();
        String pkSMS = tagReader.tagP(mod);
        this.setTextSMSString(pkSMS);
        this.smsSender.sendSms(pkSMS, number);
        Toast.makeText(this, "PK sended", Toast.LENGTH_SHORT).show();
    }

    public void openSMS(View view) {
        Intent intent = new Intent(Intent.ACTION_MAIN);
        intent.addCategory(Intent.CATEGORY_DEFAULT);
        intent.setType("vnd.android-dir/mms-sms");
        startActivity(intent);
    }

    public void sendSMS(View view) {
        String text = this.getTextSMSString();
        String number = this.getTextNumberString();
        String sms = null;
        if((text.length() > 0)
            && (number.length() > 0)
            && !tagReader.tag_predic(text)) {
            try{
                text = this.aes.encrypt(text);
            }
            catch(Exception e) {
                //varovdž"ndž" nebyl vymdž"ndž"n kldž"dž"
                Toast.makeText(this, "no AES key",
                    Toast.LENGTH_SHORT).show();
            }
            sms = this.tagReader.tagM(text);
            this.smsSender.sendSms(sms, number);
            this.setTextSMSString(sms);
            Toast.makeText(this, "SMS sended", Toast.LENGTH_SHORT).show();
        }
        else
            Toast.makeText(this, "No data", Toast.LENGTH_SHORT).show();
    }

    public void decryptSMS(View view) {
        String text = getTextSMSString();

        if((text.length() > 0)
            && (tagReader.tag_predic(text))) {
            tagReader.readSMS(text);
        }
    }

```

```

        //pokud přišel publicKey odešle se symetrický klíč zašifro-
vaný RSA
        if(tagReader.getPublicKeyString() != null) {
            rsa.initKeys(tagReader.getPublicKeyString());
            aes.initKey();

            byte[] key = rsa.encrypt(AESEncryptJava.RND_FIELD);
            String zasifKey = MainActivity.bytesToString(key);
            String sms = tagReader.tagS(zasifKey);
            String number = this.getEditTextNumberString();

            //String symK = bytesTo-
String(AESEncryptJava.RND_FIELD);
            //String sms = tagReader.tagS(symK);
            //String number = this.getEditTextNumberString();

            this.smsSender.sendSms(sms, number);
            this.setEditTextSMSString(sms);
            Toast.makeText(this, "SK sended",
Toast.LENGTH_SHORT).show();
        }
        //pokud přišel symetrický klíč
        if(tagReader.getSymetricKeyString() != null) {
            aes.key = null;
            try{
                String s = tagReader.getSymetricKeyString();

                aes.initKey(stringToBytes(rsa.decrypt(s)));
            }
            catch(Exception e) {
                Toast.makeText(this, "chyba
aes.initKey(String)", Toast.LENGTH_SHORT).show();
            }

            //this.setEditTextSMSString(tagReader.getSymetricKeyString());
            //dá info že se můžou psát sms
            Toast.makeText(this, "write sms",
Toast.LENGTH_SHORT).show();
        }
        //pokud přišla šifrovaná zpráva
        if(tagReader.getMessageString() != null) {
            String message =
aes.decrypt(tagReader.getMessageString());
            this.setEditTextSMSString(message);
        }
        //dej informaci že sms není v tagu
        else
            Toast.makeText(this, "SMS not tagged",
Toast.LENGTH_SHORT).show();
    }

    public void clear(View view) {
        this.setEditTextSMSString("");
    }

```

```
    }

    public void back(View view) {
        setContentView(R.layout.activity_main);
    }

    public void setPassword1(View view) {
        setContentView(R.layout.activity_password);
    }

    public void setPassword(View view) { }
}
//-----
public static String bytesToString(byte[] b) {
    byte[] b2 = new byte[b.length + 1];
    b2[0] = 1;
    System.arraycopy(b, 0, b2, 1, b.length);
    return new BigInteger(b2).toString(36);
}

public static byte[] stringToBytes(String s) {
    byte[] b2 = new BigInteger(s, 36).toByteArray();
    return Arrays.copyOfRange(b2, 1, b2.length);
}
}

//třída SMSSender

package com.example.smsencryptor;
import android.app.PendingIntent;
import android.content.Context;
import android.telephony.SmsManager;

public class SMSSender {

    private Context context;
    private SmsManager smsManager;

    public SMSSender(Context context){
        this.context = context;
        smsManager = SmsManager.getDefault();
    }

    public void sendSms(String text, String phoneNumber)
    {
        String destinationAddress = phoneNumber;
        PendingIntent sentIntent = null;
        PendingIntent deliveryIntent = null;
        smsManager.sendTextMessage(destinationAddress, null, text, sentIntent, deliveryIntent);
    }
}
```

```
//třída TagReader

package com.example.smsencryptor;
import android.content.Context;

public class TagReader {
    private String publicKeyString;
    private String symetricKeyString;
    private String messageString;
    private String[] tags;      //pole používaných tagů

    public TagReader(Context context) {
        this.publicKeyString = null;
        this.messageString = null;
        this.symetricKeyString = null;
        this.tags = new String[3];
        this.tags[0] = "<P"; //tag pro veřejný klíč
        this.tags[1] = "<S"; //tag pro synetrický klíč
        this.tags[2] = "<M"; //tag pro zprávu
    }

    public String cutTag(String sms) {
        return sms.substring(2);
    }

    private String tagMessage(String message, String tag) {
        String begTag = tag.substring(0, 2);
        return (begTag + message);
    }

    public String tagP(String key) {
        return tagMessage(key, this.tags[0]);
    }

    public String tagS(String message) {
        return tagMessage(message, this.tags[1]);
    }

    public String tagM(String message) {
        return tagMessage(message, this.tags[2]);
    }

    private void reset() {
        this.publicKeyString = null;
        this.symetricKeyString = null;
        this.messageString = null;
    }

    public void readSMS(String sms){
        this.reset();
        if(sms.substring(0, 2).equals(this.tags[0].substring(0, 2))) {
            this.publicKeyString = this.cutTag(sms);
        }
        else if(sms.substring(0, 2).equals(this.tags[1].substring(0, 2))) {
            this.symetricKeyString = this.cutTag(sms);
        }
        else if(sms.substring(0, 2).equals(this.tags[2].substring(0, 2))) {
            this.messageString = this.cutTag(sms);
        }
    }
}
```

```
    }

    public String getPublicKeyString() {
        return this.publicKeyString;
    }

    public String getSymetricKeyString() {
        return this.symetricKeyString;
    }

    public String getMessageString() {
        return this.messageString;
    }

    //vrátí true pokud je text v tagu
    public boolean tag_predic(String text) {
        boolean result = false;
        if(text.substring(0, 2).equals(this.tags[0].substring(0, 2))
            || (text.substring(0,
2).equals(this.tags[1].substring(0, 2)))
            || (text.substring(0,
2).equals(this.tags[2].substring(0, 2))))
            result = true;

        return result;
    }
}

// třída RSAEncrypt

package com.example.smsencryptor;

import java.math.BigInteger;
import java.security.Key;
import java.security.KeyFactory;
import java.security.KeyPair;
import java.security.KeyPairGenerator;
import java.security.NoSuchAlgorithmException;
import java.security.interfaces.RSAPublicKey;
import java.security.spec.InvalidKeySpecException;
import java.security.spec.RSAPublicKeySpec;

import javax.crypto.Cipher;

import android.content.Context;
import android.widget.Toast;

public class RSAEncrypt {

    private KeyPairGenerator kpg;
    private Key publicKey;
    private Key privateKey;
    private Context context;

    public RSAEncrypt(Context context){
        this.context = context;
        this.publicKey = null;
        this.privateKey = null;
    }
}
```

```
        this.kpg = null;
        //this.replacedKey = null;
    }

    //vytvoření klíčů
    public void initKeys() {

        try {
            kpg = KeyPairGenerator.getInstance("RSA");
        } catch (NoSuchAlgorithmException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }

        kpg.initialize(788);
        KeyPair kp = kpg.genKeyPair();
        publicKey = kp.getPublic();
        privateKey = kp.getPrivate();
    }

    //vytvoření veřejného klíče z sms
    public void initKeys(String modulus) {

        BigInteger mod = new BigInteger(modulus, 32);
        BigInteger pubExp = new BigInteger("65537");

        KeyFactory keyFactory = null;
        try {
            keyFactory = KeyFactory.getInstance("RSA");
        } catch (NoSuchAlgorithmException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
        RSAPublicKeySpec pubKeySpec = new RSAPublicKeySpec(mod, pubExp);
        RSAPublicKey key = null;
        try {
            key = (RSAPublicKey) keyFactory.generatePublic(pubKeySpec);
        } catch (InvalidKeySpecException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
        this.publicKey = key;
    }

    private RSAPublicKeySpec getPublicKeySpec() {
        KeyFactory fact = null;
        try {
            fact = KeyFactory.getInstance("RSA");
        } catch (NoSuchAlgorithmException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }

        RSAPublicKeySpec pub = null;
        try {
            pub = fact.getKeySpec(publicKey, RSAPublicKeySpec.class);
        } catch (InvalidKeySpecException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }
}
```

```
    }
    return pub;
}

private BigInteger getModulus() {
    RSAPublicKeySpec pks = getPublicKeySpec();
    BigInteger bi = pks.getModulus();
    return bi;
}

public String getModulusString() {
    BigInteger bi = this.getModulus();
    String modS = bi.toString(32);
    return modS;
}

public String encrypt(String nezasifrovanyText) {
    byte[] nezasifTextBytes = null;
    byte[] zasifTextBytes = null;
    String zasifText = null;

    try{
        nezasifTextBytes = MainActivity.stringToBytes(nezasifrovanyText);
        zasifTextBytes = encrypt(nezasifTextBytes);
        zasifText = MainActivity.bytesToString(zasifTextBytes);
    }
    catch(Exception e){
        Toast.makeText(context, "rsa.encrypt(String)",
Toast.LENGTH_SHORT).show();
    }

    return zasifText;
}

public String decrypt(String zasifrovanyText) {
    String desifrovanyText = null;
    byte[] nezasifTextBytes = null;
    byte[] zasifTextBytes = null;

    try{
        zasifTextBytes = MainActivity.stringToBytes(zasifrovanyText);
        nezasifTextBytes = decrypt(zasifTextBytes);
        desifrovanyText = MainActivity.bytesToString(nezasifTextBytes);
    }
    catch(Exception e){
        Toast.makeText(context, "rsa.decrypt(String)",
Toast.LENGTH_SHORT).show();
    }

    return desifrovanyText;
}

public byte[] encrypt(byte[] nezasifPole) {
    byte[] cipherData = null;
    Cipher cipher;
```

```
        try {
            cipher = Cipher.getInstance("RSA/None/PKCS1Padding");
            cipher.init(Cipher.ENCRYPT_MODE, publicKey);
            cipherData = cipher.doFinal(nezasifPole);
        } catch (Exception e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
        return cipherData;
    }

    public byte[] decrypt(byte[] zasifPole) {
        byte[] cipherData = null;
        Cipher cipher;

        try {
            cipher = Cipher.getInstance("RSA/None/PKCS1Padding");
            cipher.init(Cipher.DECRYPT_MODE, privateKey);
            cipherData = cipher.doFinal(zasifPole);
        } catch (Exception e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
        return cipherData;
    }
}

//třída AESEncryptJava

package com.example.smsencryptor;

import java.math.BigInteger;
import java.security.Key;
import java.security.SecureRandom;
import javax.crypto.Cipher;
import javax.crypto.spec.SecretKeySpec;
import android.content.Context;
import android.util.Base64;
import android.widget.Toast;

public class AESEncryptJava {

    private static SecureRandom SECURE_RANDOM = new SecureRandom();
    public static final byte[] RND_FIELD = SECURE_RANDOM.generateSeed(16);
    public Key key;
    private Cipher cipher;
    private Context context;

    public AESEncryptJava(Context context){
        this.context = context;
        key = null;
        cipher = null;
    }

    public void initKey() {
        key = new SecretKeySpec(AESEncryptJava.RND_FIELD, "AES");
        try{
```

```
        cipher = Cipher.getInstance("AES/ECB/PKCS5Padding");
    }
    catch(Exception e){
        Toast.makeText(context, e.getMessage(),
Toast.LENGTH_SHORT).show();
    }
}

public void initKey(String keyString) {
    byte[] original = keyString.getBytes();
    key = new SecretKeySpec(original, "AES");
}

public void initKey(byte[] bytes) {
    key = new SecretKeySpec(bytes, "AES");
    try{
        cipher = Cipher.getInstance("AES/ECB/PKCS5Padding");
    }
    catch(Exception e){
        Toast.makeText(context, e.getMessage(),
Toast.LENGTH_SHORT).show();
    }
}

public String getKeyString() {
    BigInteger n = new BigInteger(1, key.getEncoded());
    String s = n.toString(32);
    return s;
}

public String encrypt(String nezasifrovanyText){
    byte[] nezasifTextBytes = null;
    byte[] zasifTextBytes = null;
    String zasifText = null;

    try{
        nezasifTextBytes = nezasifrovanyText.getBytes("Unicode");
        zasifTextBytes = encrypt(nezasifTextBytes);
        zasifText = Base64.encodeToString(zasifTextBytes, 0);
    }
    catch(Exception e){
        Toast.makeText(context, "aes.encrypt(String)",
Toast.LENGTH_SHORT).show();
    }

    return zasifText;
}

public String decrypt(String zasifrovanyText){
    String desifrovanyText = null;
    byte[] nezasifTextBytes = null;
    byte[] zasifTextBytes = null;

    try{
        zasifTextBytes = Base64.decode(zasifrovanyText, 0);
        nezasifTextBytes = decrypt(zasifTextBytes);
        desifrovanyText = new String(nezasifTextBytes, "Unicode");
    }
    catch(Exception e){
```

```
        Toast.makeText(context, "aes.decrypt(String)",
Toast.LENGTH_SHORT).show();
    }

    return desifrovanyText;
}

private byte[] encrypt(byte[] nezasifrovanyTextBytes){
    byte[] result = null;
    try{
        cipher.init(Cipher.ENCRYPT_MODE, key);
        result = cipher.doFinal(nezasifrovanyTextBytes);
    }
    catch(Exception e){
        Toast.makeText(context, "aes.encrypt(byte[])",
Toast.LENGTH_SHORT).show();
    }

    return result;
}

private byte[] decrypt(byte[] zasifrovanyTextBytes){
    byte[] result = null;
    try{
        cipher.init(Cipher.DECRYPT_MODE, key);
        result = cipher.doFinal(zasifrovanyTextBytes);
    }
    catch(Exception e)
    {
        Toast.makeText(context, "aes.decrypt(byte[])",
Toast.LENGTH_LONG).show();
    }

    return result;
}
}
```