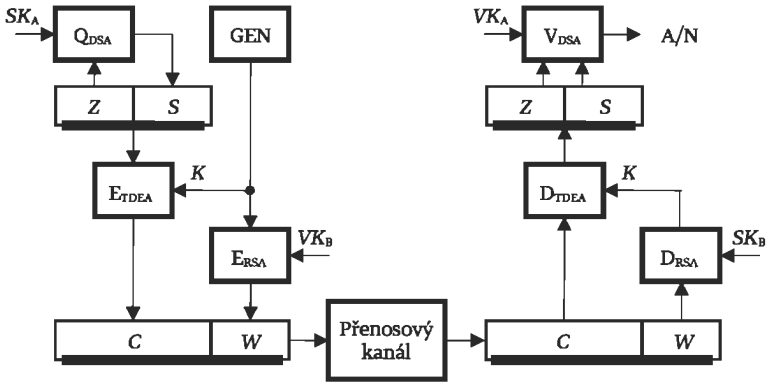
elektronické pošty šifrovat, podepisovat nebo šifrovat a podepisovat. My si popíšeme posledně uvedenou variantu.

Předpokládáme, že původce A má k dispozici ověřený certifikát veřejného šifrovacího klíče VK_B adresáta B, certifikát svého veřejného ověřovacího klíče VK_A a samozřejmě zná odpovídající soukromý podepisovací klíč SK_A. Dále předpokládáme, že původce použije kryptografickou kombinaci, která je pro poštovní klienty povinná, tj. použije kombinaci, kdy podpis se uskuteční algoritmem DSA za pomoci hešovací funkce SHA-1 a klíč symetrického kryptosystému bude adresátovi bezpečně doručen prostřednictvím kryptosystému RSA. Kryptografické zabezpečení zprávy Z pro uvedené předpoklady ilustruje obr. 6.6. Schéma vlevo popisuje činnost původce a schéma vpravo ilustruje činnost adresáta elektronického dopisu.



Obr. 6.6: Princip zabezpečení elektronické pošty.

Zpráva Z je nejprve soukromým klíčem SK_A původce pomocí algoritmu DSA podepsána. Podpis $S = Q_{DSA}(Z, SK_A)$ je připojen ke zprávě, čímž vznikne podepsaná zpráva (Z, S). Původce nyní pomocí generátoru náhodných čísel GEN vygeneruje náhodný klíč K pro zašifrování podepsané zprávy. Dejme tomu, že původce se rozhodl použít algoritmus blokové šifry TDEA. Podepsaná zpráva (Z, S) je touto blokovou šifrou v režimu CBC zašifrována do podoby kryptogramu $C = E_{TDEA}[(Z, S), K]$. Dále je klíč K kryptosystémem RSA pomocí veřejného klíče adresáta zašifrován do podoby $W = E_{RSA}(K, VK_B)$.

Dvojice (C, W) je spolu s dalšími potřebnými informacemi (certifikát veřejného klíče původce, identifikátory použitých algoritmů a inicializační vektor) odeslána adresátovi.

Příjemce si nejprve ověří platnost certifikátu původce a z něho získá veřejný ověřovací klíč původce VK_A. Dále pomocí svého soukromého dešifrovacího klíče dešifruje kryptogram W, čímž získá klíč $K = D_{RSA}(W, SK_B)$. Pomocí tohoto klíče dešifruje kryptogram C, čímž získá podepsanou zprávu $(Z, S) = D_{TDEA}(C, K)$. Autentičnost přijaté zprávy ověří pomocí veřejného klíče původce. Pokud je výstupem ověřovací funkce $V_{DSA}(Z, S, VK_A)$ tvrzení, že zpráva Z je autentická, adresát přijatý elektronický dopis akceptuje.

Z popisu je zřejmé, že se jedná o hybridní kryptografický systém, kde jsou zkombinovány symetrický a asymetrické kryptosystémy. Symetrický kryptosystém zajišťuje vysokou rychlost šifrování a dešifrování, kryptosystém RSA zajišťuje bezpečný transport klíče symetrického kryptosystému a kryptosystém DSA zajišťuje autentičnost zprávy spolu s neodmítnutelností odpovědnosti původce.

Poznamenáváme, že výše uvedený popis je poněkud zjednodušený. V popisu je vynecháno generování inicializačního vektoru IV pro šifrovací režim CBC a rovněž je vynecháno zarovnání zprávy na celistvý násobek délky bloku použité šifry. Ohledně použité šifry je zapotřebí vysvětlit, že šifra označovaná TDEA („Triple Data Encryption Algorithm" nebo nesprávně 3DES) je trojnásobné vykonání šifrovacího algoritmu DES („Data Encryption Standard") se třemi různými klíči. Šifra DES je již zastaralá a je nahrazována standardem AES, který jsme si popisovali ve 3. kapitole. Přesto se s algoritmem DES v praxi stále ještě setkáváme. Šifra DES má délku bloku 64 bitů a délku klíče 56 bitů. Jak již víme, taková délka klíče je nedostatečná. Ke zvýšení bezpečnosti při současném využití jinak oblíbeného a rozšířeného standardu DES se zavedl právě algoritmus TDEA. Princip spočívá v tom, že pro algoritmus TDEA se stanovuje klíč $K = K_1 \parallel K_2 \parallel K_3$, což je náhodný blok o délce 3·56 = 168 bitů. Každý blok B_i zprávy Z se šifrou DES transformuje třikrát. Pro odpovídající blok kryptogramu platí, že $C_i = E_{TDEA}(B_i, K) = E_{DES}\{D_{DES}[E_{DES}(B_i, K_1), K_2], K_3\}$. Znamená to, že blok B_i se nejprve zašifruje klíčem K_1 do podoby kryptogramu $C_{i1} = E_{DES}(B_i, K_1)$. Tento kryptogram se dešifrovací transformací klíčem K_2 převede do podoby kryptogramu $C_{i2} = D_{DES}(C_{i1}, K_2)$ a poslední transformací získáme výsledný blok kryptogramu $C_i = E_{DES}(C_{i2}, K_3)$. Šifrování a dešifrování je sice zhruba třikrát pomalejší, ale významně se tím zvýšila bezpečnost. Dešifrování u TDEA se provádí inverzně k šifrování, tj. platí, že $B_i = D_{TDEA}(C_i, K) = D_{DES}\{E_{DES}[D_{DES}(C_i, K_3), K_2], K_1\}$.

Vzhledem k úspěchům kryptoanalýzy, kdy prakticky každý navržený kryptosystém byl nakonec překonán, se v minulosti řada kryptologů začala domnívat, že nepřekonatelný kryptosystém ani neexistuje. Tento názor například v roce 1843 formuloval amatérský kryptolog a spisovatel E. A. Poe, který ve své povídce [3] (s. 105) ústy hlavního hrdiny příběhu vyslovuje otázku, zda je lidský důmysl vůbec schopen sestrojit takovou šifru, kterou by lidský důmysl opět nebyl schopen vyřešit. Definitivní odpověď na tuto otázku přišla až v roce 1949 a jejím autorem byl americký matematik C. E. Shannon.

Pan Shannon ve své práci [4] uvádí, že utajovací kryptosystém může poskytnout tři stupně důvěrnosti:

- maximální důvěrnost („perfect secrecy“),
- redukovanou důvěrnost („ideal secrecy“),
- minimální důvěrnost („practical secrecy“).

Uvedené stupně důvěrnosti zpráv jsou definovány pomocí pojmů z teorie informace. My si je vysvětlíme jednodušeji a názorněji pomocí analýzy šifrovací a dešifrovací funkce.

Šifrování E jsme definovali jako prostou funkci, která je určena šifrovacím klíčem K_E. Tato funkce každé zprávě Z přiřadí číslo C, které se nazývá kryptogram. Formálně jsme funkci šifrování vyjádřili následovně:

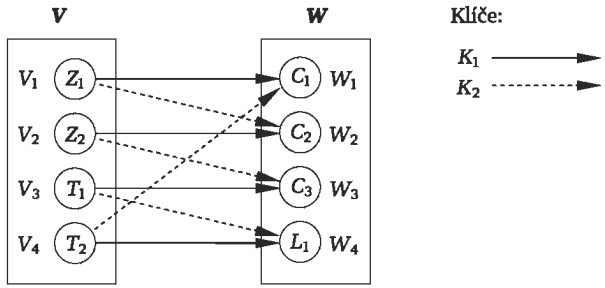
$$C = E(Z, K_E).$$

Nyní si definici funkce šifrování rozšíříme a zpřesníme. Jak již bylo uvedeno, šifrovací klíč K_E je považován za parametr, a tak pro konkrétní hodnotu klíče lze šifrování považovat za funkci jedné proměnné, kdy je vstupní hodnotě Z přiřazena hodnota výstupu C. Množinou vstupů $\boldsymbol{V}$ šifrovací funkce E jsou čísla V (tzv. vzory), s nimiž může daný kryptografický systém pracovat. Pokud je ve vzoru $V \in \boldsymbol{V}$ zakódována informace, pak se jedná o zprávu Z. Vzor, v němž není zakódována žádná informace, nazveme prázdným vzorem T. Množinu výstupů $\boldsymbol{W}$ funkce E tvoří hodnoty W (tzv. obrazy) všech vzorů. Obraz $W \in \boldsymbol{W}$, který je alespoň pro jednu hodnotu klíče K_E přiřazen nějaké zprávě, nazýváme kryptogram C. Ostatní obrazy nazveme prázdným obrazem L.

Předpokládáme, že počet prvků množin $\boldsymbol{V}$ i $\boldsymbol{W}$ je roven číslu m a připomínáme, že funkce šifrování je prostá funkce. Máme-li množinu šifrovacích klíčů $\boldsymbol{K}$, potom pro každý klíč $K_E \in \boldsymbol{K}$ platí, že pro každou dvojici vzorů $V_1 \neq V_2$ lze psát $E(V_1, K_E) \neq E(V_2, K_E)$. Každý klíč tedy přiřazuje každému vzoru z m možných jeden

unikátní obraz z m možných. Pokud tvoříme konkrétní šifrovací transformaci, zvolíme si první vzor, a tomu přiřadíme jeden z m obrazů. Druhému vzoru pak přiřadíme jeden z $(m - 1)$ zbývajících obrazů atd. Potom pro maximální počet přiřazení, tj. pro maximální počet klíčů K_{Max}, platí $K_{Max} = m!$ V prakticky používaných systémech bývá počet možných klíčů podstatně menší.

Výše uvedené definice ilustruje obrázek 1.5. Množinu vstupů $\boldsymbol{V}$ funkce šifrování tvoří čtyři vzory V_1 až V_4. Informaci nesou vzory V_1 a V_2 (jsou označeny jako zprávy Z_1 a Z_2) a zbývající dva vzory nemají v daném jazyce nebo kódu význam. Ty jsou označeny jako prázdné vzory T_1 a T_2. V kryptosystému jsou definovány dva klíče. Funkce šifrování je pro první klíč K_1 definována plnými čarami s šipkami a pro druhý klíč K_2 je definována přerušovanými čarami s šipkami. Množina obrazů $\boldsymbol{W}$ je definována rovněž čtyřmi prvky, protože šifrovací funkce musí pro každý daný klíč kvůli jednoznačnosti dešifrování přiřadit každému možnému vzoru unikátní obraz. V našem příkladě v množině obrazů existují tři kryptogramy C_1 až C_3 a jeden prázdný obraz L_1.



Obr. 1.5: Příklad funkce šifrování.

Nyní definujme valenci *Val* obrazu W_i jako počet zpráv, které vzniknou dešifrováním W_i všemi možnými klíči. Na našem obrázku například $Val(W_1) = 1$, protože dešifrováním obrazu W_1 pomocí klíče K_1 vznikne V_1 a dešifrováním pomocí klíče K_2 vznikne V_4. Protože zprávou je jen V_1, valence W_1 je rovna jedné. Platí, že valence kryptogramu je alespoň 1 a nanejvýše Min{N, K}, kde N je celkový počet zpráv a K je celkový počet klíčů. Obraz s valencí rovnou nule je prázdný obraz.

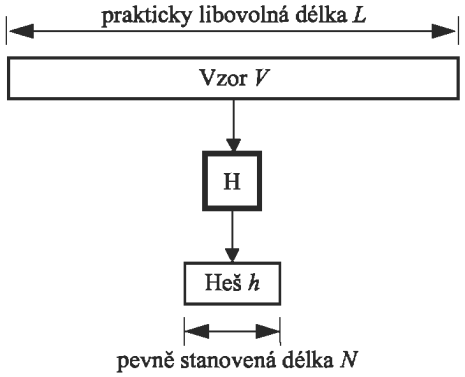
Dále definujme veličinu M jako minimální valenci ze všech kryptogramů. V našem příkladě je valence všech kryptogramů rovna hodnotě 1, 2, a 1 a tak

Jednosměrné funkce používané v kryptografii můžeme klasifikovat na:

- funkce s pevnou délkou výstupu,
- funkce s volitelnou délkou výstupu.

Jednosměrné funkce s pevnou délkou výstupu přiřazují vzoru o prakticky libovolné bitové délce určitý obraz, který má pevně stanovenou délku. Tento typ funkce se obvykle nazývá hešovací funkce („hash function") a používá se k tomu, abychom zprávě přiřadili relativně krátký bitový řetězec o pevné délce, který slouží jako reprezentant dané zprávy. V případě funkce s volitelnou délkou výstupu je bitová délka obrazu volitelná. Délka výstupu může být obecně buď kratší než vstup (kompresní funkce), stejná (ekvivalentní funkce), nebo delší (expanzní funkce). V kryptografické praxi se z uvedených tří možností nejvíce potkáme s jednosměrnými expanzními funkcemi. Tato třída funkcí se obvykle používá k odvozování klíčů a ke generování delších bitových posloupností z relativně krátkých posloupností.

S hešovacími funkcemi (např. SHA-1, MD5 atd.) se v praxi potkáváme velmi často a jejich teorie je poměrně dobře propracována. V případě jednosměrných expanzních funkcí tomu tak není. Asi nejznámějšími reprezentanty těchto funkcí je funkce MGF (viz 4. kapitola) a KDF (viz 5. kapitola). V dalším také uvidíme, že za jednosměrnou expanzní funkci můžeme považovat i generátory pseudonáhodné posloupnosti (viz následující podkapitola). Zbytek této podkapitoly proto budeme věnovat popisu hešovacích funkcí a popis jednotlivých reprezentantů jednosměrných expanzních funkcí si ponecháme do příslušných kapitol.

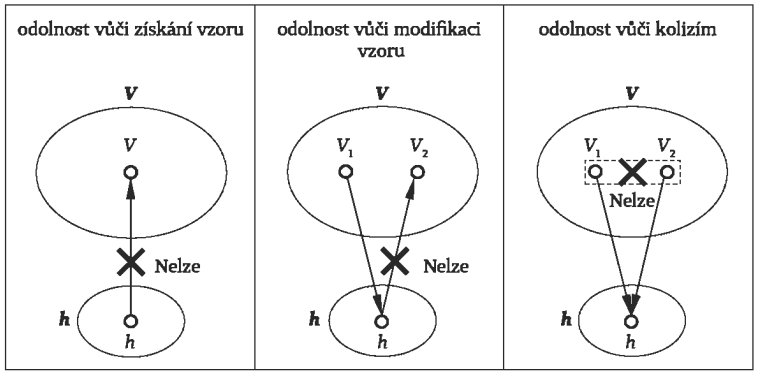


Obr. 2.1: Účel hešovací funkce.

Hešovací funkce se nejčastěji využívají v autentizačních kryptostémech. Hlavním účelem těchto funkcí (viz obr. 2.1) je vytvářet reprezentanty zpráv o konstantní délce, tzv. heše. Vzorem pro hešovací funkci H je binární posloupnost V (nejčastěji nějaká zpráva) o prakticky libovolné délce L bitů a obrazem funkce je binární posloupnost h (tzv. heš) o pevně stanovené délce N (typicky 128 až 512 bitů). Vzhledem k délkám vzorů L a hešů N zcela pochopitelně platí, že počet prvků množiny vzorů je mnohonásobně větší než počet prvků množiny hešů. Důsledkem této skutečnosti je fakt, že každý heš musí reprezentovat více vzorů, tj. obrazem těchto vzorů je společný heš. Tento jev se nazývá kolize vzorů.

Hešovací funkce je obvykle funkce, od které se požaduje:

- odolnost vůči získání vzoru („preimage resistance"),
- odolnost vůči modifikaci vzoru („2nd-preimage resistance"),
- odolnost vůči kolizím („collision resistance").



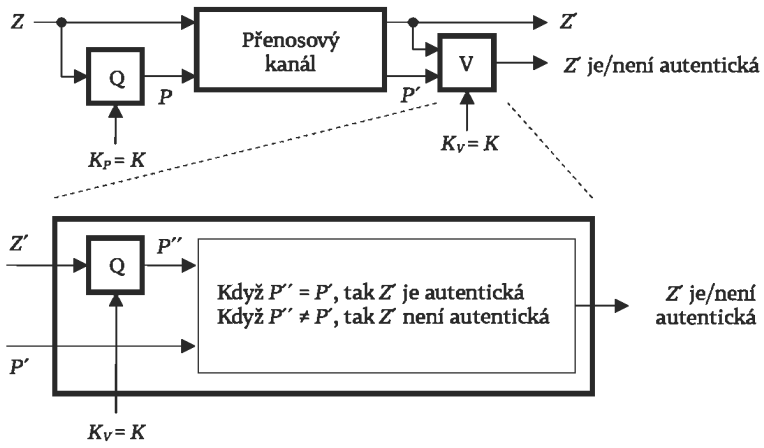
Obr. 2.2: Požadavky na hešovací funkci.

Uvedené požadavky ilustruje trojice diagramů na obr. 2.2. Horní elipsa reprezentuje množinu $\boldsymbol{V}$ všech vzorů („preimage") a dolní elipsa reprezentuje množinu $\boldsymbol{h}$ všech hešů („image"). Odolností vůči získání vzoru (tzv. jednosměrnost) se rozumí požadavek, aby pro dané h bylo prakticky nemožné nalézt takové V, pro které by platilo $H(V) = h$. Vzhledem k tomu, že počet všech hešů je roven 2^N,

V předchozích odstavcích jsme si popsali nejčastěji používané režimy provozu blokových šifer. V této souvislosti je zapotřebí poznamenat, že provozních režimů existuje mnohem větší množství. V této knize se ještě setkáme s režimem CFB a CCM (viz 6. kapitola). Pro konkrétní scénář nasazení kryptografického systému je zapotřebí vybrat ten nejvhodnější.

3.3 AUTENTIZACE SYMETRICKÝMI KRYPTOSYSTÉMY

V této kapitole, která je věnována symetrickým kryptosystémům, jsme se doposud zabývali třídou utajovacích kryptosystémů. Nyní nadešel čas, abychom si vysvětlili problematiku symetrických autentizačních kryptosystémů. Jak již bylo uvedeno, autentizace slouží k tomu, aby si adresát mohl ověřit původ zprávy (kdo je původcem zprávy, kdy zpráva vznikla atd.). Princip je takový, že původce ke zprávě připojuje dodatečná data (tzv. pečeť), které adresátovi umožní původ zprávy ověřit. Pečeti se v případě symetrických autentizačních systémů nazývají různě – například MAC („Message Authentication Code", MIC („Message Integrity Check") nebo HMAC („Hashed MAC").



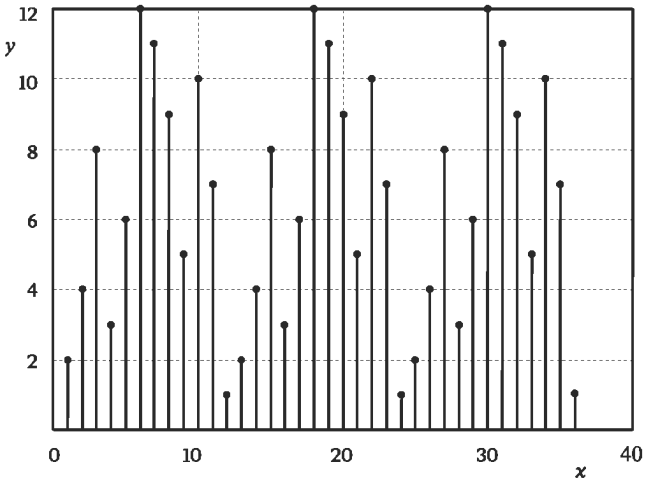
Obr. 3.23: Symetrický autentizační kryptosystém.

Na obr. 3.23 je uvedeno obecné schéma symetrického autentizačního kryptosystému. Pečeť se obecně tvoří pečetícím klíčem K_P a ověřuje ověřovacím klíčem K_V, avšak v praxi platí, že oba klíče jsou stejné, tj. $K_P = K_V = K$. Tajný klíč K budeme nazývat autentizační klíč. Na straně původce se k dané zprávě Z odvodí pomocí pečetící funkce Q a autentizačního klíče K pečeť $P = Q(Z, K)$. Dvojice (Z, P), tj. autentizovaná zpráva, je odeslána adresátovi. Adresát obecně přijme dvojici (Z', P'), protože útočník se mohl pokusit o modifikaci zprávy v průběhu jejího přenosu. Následně adresát zjistí, jaká pečeť přísluší Z', tj. vypočítá $P'' = Q(Z', K)$. Pokud adresátem vypočítaná pečeť P'' je rovna přijaté pečeti P', pak je zpráva Z' považována za autentickou. Vychází se z toho, že pro libovolnou zprávu Z' by měl být schopen určit správnou pečeť P' pouze původce a adresát.

Jak již bylo zmíněno, existuje celá řada typů pečetí. My se seznámíme asi s nejrozšířenějším typem pečetí HMAC a CMAC. Pečeti typu HMAC („Hash-based Message Authentication Code") se vypočítávají pomocí hešovací funkce H. V současné době pečeť HMAC definuje internetový standard RFC-2104 [18] a americký standard FIPS-PUB-198 [19].

K výpočtu HMAC je zapotřebí nějaké hešovací funkce H. Základními parametry hešovací funkce pro výpočet HMAC je délka bloků r a délka heše l. Bloky se zde rozumí části, na které je vstup do hešovací funkce rozdělován. Například u SHA-1 jsme si uvedli, že délka bloků r = 512 b = 64 B a délka heše l = 160 b = 20 B. V obou výše uvedených standardech se operuje s bajty, a tak v dalším budou veličiny r a l uváděny v bajtech.

K výpočtu pečeti je zapotřebí autentizační klíč K. Pro délku $|K|$ tohoto klíče by podle [18] mělo platit $|K| \geq l$ a podle [19] by mělo platit $|K| \geq l/2$. Na tomto požadavku vidíme, že různé standardy mohou jednu a tutéž věc řešit dosti rozdílně. Je to dáno tím, že hodnotitelé bezpečnosti kryptosystémů používají buď různé metody hodnocení, nebo předpokládají odlišné scénáře nasazení jimi hodnoceného kryptosystému. Provozovatel kryptosystému musí v takovém případě porozumět rozdílům mezi standardy a pro implementaci vybrat standard, který odpovídá scénáři nasazení jeho kryptosystému, nebo standard, jehož metoda hodnocení bezpečnosti se mu jeví důvěryhodnější. Vybraného standardu je zapotřebí se držet a nesnažit se kombinovat vybrané části více standardů. Kombinováním více standardů totiž může vzniknout kryptosystém, který bude obsahovat slabiny navíc.



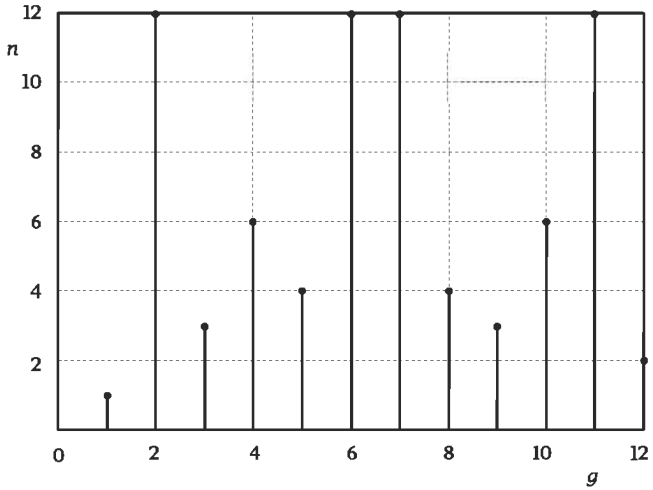
Obr. 4.10: Ilustrace cyklické grupy na závislosti $y = 2^x \bmod 13$.

V kryptosystémech typu DL se používají cyklické grupy založené na vztahu $y = g^x \bmod p$, kde y, g a x jsou celá kladná čísla z intervalu $\langle 1, p-1\rangle$ a p je prvočíslo. V takovém případě počet prvků, které generátor g vygeneruje, je nějakým součinem prvočíselných faktorů čísla $(p-1)$. V našem úvodním příkladě můžeme číslo $(p-1) = 12$ faktorizovat na součin prvočísel $2^2 \cdot 3$, tj. v množině $\boldsymbol{G}_{12}$ nalezneme generátory, které dokážou vygenerovat množinu o $n = 12, 6, 4, 3, 2$ a 1 prvcích. Číslo n se nazývá řád generátoru nebo také řád vygenerované množiny a množinu řádu n budeme značit $\boldsymbol{G}_n$. Například číslo 3 dokáže vygenerovat posloupnost (3, 9, 1), tj. dokáže vygenerovat tříprvkovou množinu {1, 3, 9}. Pro generátor g a jeho řád n platí, že $g^n \bmod p = 1$. Grupy, jejichž řád je menší nežli $(p-1)$, nazýváme také někdy podgrupy. Na obr. 4.11 jsou pro náš ilustrativní příklad uvedeny řády grup vygenerovaných různými generátory.

Z obrázku vidíme, že například $g = 2, 6, 7$ a 11 dokážou vygenerovat celou grupu $\boldsymbol{G}_{12} = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12\}$. Ostatní generátory dokážou vygenerovat podgrupy $\boldsymbol{G}_n$, kde $n = 6, 4, 3, 2$ a 1. Obecně platí, že $\boldsymbol{G}_1 = \{1\}$ a $\boldsymbol{G}_2 = \{1, p-1\}$.

Nyní můžeme přejít k popisu kryptosystému typu DL. Budoucí majitel kryptosystému typu DL musí nejprve vygenerovat parametry p, g, x a y. Veřejné prvočíslo p musí mít z bezpečnostních důvodů tu vlastnost, že hodnota $(p-1)$

musí být násobkem dostatečně velkého prvočísla q, které je pak řádem použité podgrupy. V současné době se jako minimálně bezpečné doporučuje používat prvočíslo q o délce $N = 224$ bitů a prvočíslo p by mělo mít délku $L = 2048$ bitů [30]. Podle uvedeného standardu se vygeneruje prvočíslo q a poté i prvočíslo p o požadovaných délkách a s vlastností, že $(p-1)$ je celistvým násobkem čísla q. Pokud pro ilustraci postupu zvolíme $q = 11$, pak vhodným prvočíslem p bude číslo 23, neboť $(p-1) = (23-1) = 22 = 2 \cdot 11 = 2 \cdot q$.



Obr. 4.11: Závislost řádu n grupy na hodnotě generátoru g pro $y = g^x \bmod 13$.

DL kryptosystémy se konstruují na podgrupě $\boldsymbol{G}_q$. Dalším krokem je proto určení generátoru takovéto podgrupy. Jak již bylo uvedeno, generátor $g \in \langle 2, p-1\rangle$ podgrupy $\boldsymbol{G}_q$ je celé číslo, které má tu vlastnost, že pokud postupně použijeme všechny možné exponenty $x \in \langle 1, q\rangle$, pak mocněním $y = g^x \bmod p$ získáme celkem q různých hodnot mocnin. Generátor se ve zmiňovaném standardu určuje následovně. Vypočítá se hodnota $e = (p-1)/q$, náhodně se zvolí $h \in \langle 2, p-2\rangle$, a pokud $a = h^e \bmod p \neq 1$, pak $g = a$. V opačném případě zvolíme jinou hodnotu h a postup opakujeme. V našem ilustrativním příkladě je e rovno hodnotě $(23-1)/11 = 2$ a náhodně zvolíme $h = 2$. Zjistíme, že $a = 2^2 \bmod 23 = 4 \neq 1$, a tak generátor $g = a = 4$. Tímto způsobem jsme vygenerovali obecné veřejné parametry pro DL kryptosystém, tj. vygenerovali jsme $p = 23$, $q = 11$ a $g = 4$.

klad v případě čipové karty se často sleduje stabilita vnějšího napájení. Důvodem je skutečnost, že procesor čipové karty může v případě kolísajícího napětí nesprávně interpretovat nějakou instrukci svého programu. To může vést k odskoku z hlavního programu do diagnostických podprogramů karty, které pak útočníkovi umožní výpis obsahu paměti karty, a tudíž také zjištění hodnoty uloženého klíče. Z tohoto důvodu se změny hodnoty napájecího napětí považují za pokus o útok a karta v takovém případě provede výmaz uložených hodnot. Přenosná paměťová úložiště se nejvíce používají pro transporty klíčů symetrických kryptosystémů (tzv. „key loader"). Používají je však i majitelé asymetrických kryptosystémů pro bezpečné uložení a přenos soukromých klíčů svých kryptosystémů. Nevýhodou tohoto typu transportu je pomalý přenos.

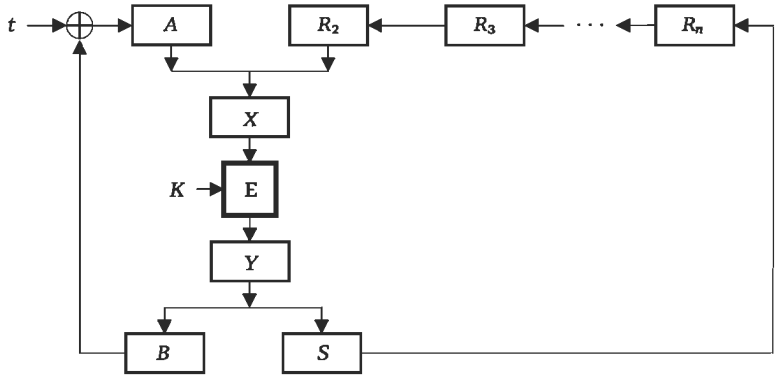
Techniky transportu klíče pomocí symetrického kryptosystému jsou založeny na tom, že transportovaný klíč k je zašifrován klíčem K pomocí symetrického kryptosystému do podoby kryptogramu C a tento kryptogram je odeslán oprávněné straně. Ta jej pomocí klíče K dešifruje a získá tak klíč k. Pro tento účel jsou v současné době standardizovány tzv. techniky zabalení klíče („key wrapping"). Zmíněné techniky zajišťují důvěrnost i autentičnost transportovaného klíče. My si zde popíšeme techniku zabalení klíče podle návrhu standardu [35] ve variantě s blokovou šifrou AES.

U popisované techniky (viz obr. 5.3) jsou základními datovými jednotkami bloky o délce 64 bitů. Tyto bloky budeme nazývat slova. Transportovaný klíč se rozdělí na $(n-1)$ slov, které se umístí do registrů R_2 až R_n, přičemž $n \geq 3$. Z hodnoty n plyne, že minimální délka transportovaného klíče jsou 2 slova, tj. 128 bitů. Do registru A se umístí kontrolní slovo M, kterým příjemce ověřuje autentičnost transportovaného klíče. Ve standardu je stanoveno, že $M = \text{A6A6A6A6A6A6A6A6}_{16}$. S obsahem registrů A a R_2 až R_n se provede $s = 6 \cdot (n-1)$ následujících iterací:

1. $Y = \text{E}(X, K)$, kde $X = A \,||\, R_2$ a E je šifrování algoritmem AES v režimu ECB.
2. $B = \text{MSB}^{64}(Y)$, $S = \text{LSB}^{64}(Y)$.
3. $A = B \oplus t$, kde $t \in \{1, 2, ..., s\}$ je pořadové číslo iterace. Číslo t je dlouhé 64 bitů.
4. $R_i = R_{i+1}$, kde $i = 2, 3, ..., (n-1)$. $R_n = S$.

Popsanou iteraci ilustruje obr. 5.3. Při první iteraci se v registru A nachází kontrolní slovo M, v registru R_2 se nachází první slovo transportovaného klíče k a v registrech R_3 až R_n se nacházejí zbývající slova klíče k. V prvním kroku iterace se obsah registrů A a R_2 zřetězí do podoby 128 bitů dlouhého řetězce X, který se pomocí transportního klíče K zašifruje algoritmem AES do podoby kryptogramu Y.

Ve druhém kroku se 64 nejvíce významných bitů (MSB) kryptogramu Y (tj. prakticky levá polovina Y) stane slovem B a 64 nejméně významných bitů (LSB) kryptogramu Y (tj. prakticky pravá polovina Y) se stane slovem S. Ve třetím kroku se změní obsah registru A na hodnotu $B \oplus t$, kde t je 64 bitů dlouhé pořadové číslo iterace. Ve čtvrtém kroku se obsahy registrů R_3 až R_n posunou o jeden registr vlevo a na uprázdněné místo v registru R_n se zapíše slovo S.

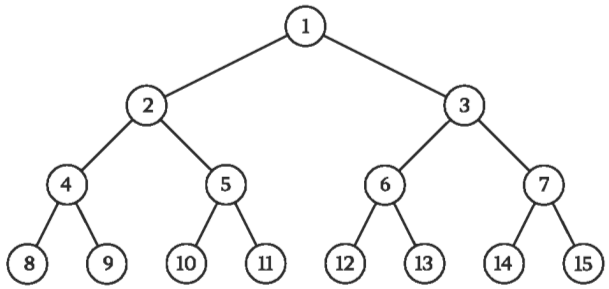


Obr. 5.3: Jednotlivá iterace algoritmu zabalení klíče.

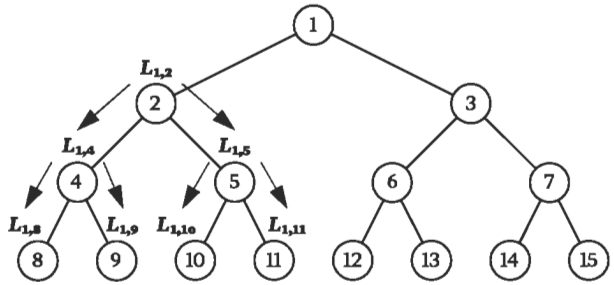
Postupným vykonáním všech s iterací se dosáhne toho, že obsah registru A bude velmi složitým způsobem záviset na výchozím kontrolním slově M a na hodnotách všech slov transportovaného klíče k. Obsah registru A proto slouží jako autentizační kód zabaleného klíče. Obsah registrů R_2 až R_n na počátku zabalovacího algoritmu obsahoval jednotlivá slova transportovaného klíče k, a tak tyto registry na konci algoritmu obsahují zašifrovaná slova transportovaného klíče k. Zřetězený obsah registrů A a R_2 až R_n je zabaleným klíčem k.

Zabalený klíč je přenesen k adresátovi. Ten má k dispozici tajný transportní klíč K a může tak provést s inverzních iterací. Pokud se po jejich provedení bude v registru A nacházet kontrolní slovo M, zabalený klíč nebyl během svého přenosu od odesílatele pozměněn a v registrech R_2 až R_n se nacházejí jednotlivá slova transportovaného klíče k.

Rozbalení klíče si můžeme ukázat opět na obr. 5.3. Pokud kromě šipek u parametrů t a K otočíme směr těchto šipek, můžeme se ze stavu, který byl



Obr. 7.2: Úplný binární strom.

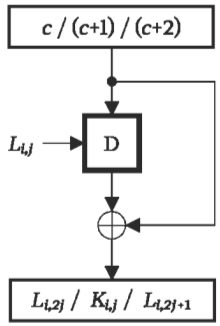


Obr. 7.3: Hierarchie semen v úplném binárním stromu.

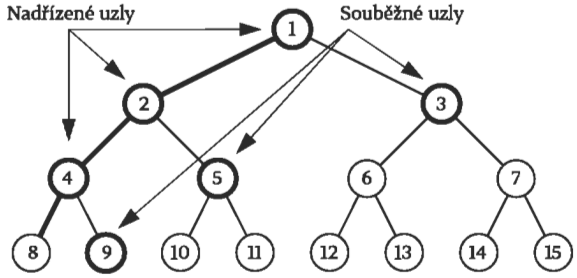
Jednosměrná expanzní funkce AES-G3 je na obr. 7.4. Veličina c je 128 bitů dlouhá konstanta, které je určena standardem AACS a operátor D označuje algoritmus blokové šifry AES v režimu dešifrování ECB. V prvním běhu algoritmu AES se odvodí semeno $L_{i,2j} = D(c, L_{i,j}) \oplus c$. Následně se konstanta c inkrementuje o 1 a ve druhém běhu algoritmu se odvodí klíč $K_{i,j} = D(c+1, L_{i,j}) \oplus (c+1)$. V posledním třetím běhu je na vstupu dešifrátoru D hodnota $(c+2)$ a výstupem je semeno $L_{i,2j+1} = D(c+2, L_{i,j}) \oplus (c+2)$.

K určení klíčů pro jednotlivé přehrávače definujeme nadřízené a mimoběžné uzly. Pro list k (tj. pro přehrávač reprezentovaný listem k) jsou nadřízenými uzly takové uzly, které se nacházejí v cestě od uzlu k ke kořenu. Množinu nadřízených uzlů označme $\boldsymbol{N}$. Mimoběžnými uzly pro list k jsou takové uzly, které mu nadřízené nejsou. Jedná se tedy prakticky o všechny uzly, které nejsou

součástí cesty z uzlu k ke kořeni stromu. Mimoběžné uzly, které sousedí (tj. jsou spojeny jedinou hranou) s některým z nadřízených uzlů, nazveme souběžné uzly a množinu takovýchto uzlů budeme značit $\boldsymbol{M}$. Například pro uzel $k = 8$ je $\boldsymbol{N} = \{1, 2, 4\}$ a $\boldsymbol{M} = \{3, 5, 9\}$ (viz obr. 7.5).



Obr. 7.4: Jednosměrná expanzní funkce AES-G3.



Obr. 7.5: Nadřízené a souběžné uzly.

Při výrobě přehrávače se do jeho paměti uloží všechna semena $L_{i,j}$, kde $i \in \boldsymbol{N}$ a $j \in \boldsymbol{M}$, přičemž $j \geq 2 \cdot i$. Potom například přehrávač odpovídající uzlu 8 zná semena z množiny $\{L_{1,3}, L_{1,5}, L_{1,9}, L_{2,5}, L_{2,9}, L_{4,9}\}$. Pomocí funkce AES-G3 si z $L_{1,3}$ ještě dokáže odvodit semena $\{L_{1,6}, L_{1,7}, L_{1,12}, L_{1,13}, L_{1,14}, L_{1,15}\}$, z $L_{1,5}$ odvodí $\{L_{1,10}, L_{1,11}\}$ a z $L_{2,5}$ odvodí $\{L_{2,10}, L_{2,11}\}$. Z těchto jemu dostupných semen si tedy uvedený přehrávač může pomocí funkce AES-G3 odvodit všechny jemu při-