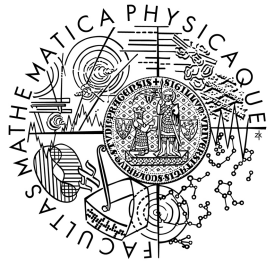
Univerzita Karlova v Praze
Matematicko-fyzikální fakulta

BAKALÁŘSKÁ PRÁCE



Marie Švarcová

Důkazy bezpečnosti schémat symetrické kryptografie

Katedra algebry

Vedoucí bakalářské práce: RNDr. Bohuslav Rudolf

Studijní program: matematika, matematické metody informační bezpečnosti

2010

Ráda bych tímto poděkovala všem, kteří mi během psaní této práce pomáhali, zejména pak vedoucímu práce RNDr. Bohuslavu Rudolfovi za jeho čas a podnětné připomínky.

Prohlašuji, že jsem svou bakalářskou práci napsala samostatně a výhradně s použitím citovaných pramenů. Souhlasím se zapůjčováním práce a jejím zveřejňováním.

V Praze dne 26. května 2010 Marie Švarcová

Obsah

Název práce: Důkazy bezpečnosti schémat symetrické kryptografie
Autor: Marie Švarcová
Katedra (ústav): Katedra algebry
Vedoucí bakalářské práce: RNDr. Bohuslav Rudolf
e-mail vedoucího: b.rudolf@nbu.cz

Abstrakt: V předložené práci zkoumáme techniky důkazů bezpečnosti schémat symetrické kryptografie se zaměřením na schémata šifrování s ochranou integrity. Výsledky z této oblasti jsou v jednotlivých kapitolách vždy představeny a dokázány. Začínáme studiem bezpečnostních pojmů symetrické kryptografie a jejich vzájemnými vztahy. Dále následuje studium generických schémat šifrování s ochranou integrity a jejich bezpečnosti a zavedení hašovacích funkcí s klíčem a schématu NMAC. Posledním studovaným tématem jsou pak schémata šifrování s ochranou integrity s připojenými daty. V závěru pak uvedené důkazy popisujeme obecně a přinášíme jednotící pohled na předvedené důkazové techniky.

Klíčová slova: symetrická kryptografie, schémata šifrování s ochranou integrity, důkazy bezpečnosti

Title: Security Proofs in Symmetric Cryptography
Author: Marie Švarcová
Department: Department of Algebra
Supervisor: RNDr. Bohuslav Rudolf
Supervisor's e-mail address: b.rudolf@nbu.cz

Abstract: In the present work we investigate into security proofs techniques in symmetric cryptography with aim at authenticated encryption schemes. The results of this area are in each chapter shown and proven. We begin with studying security notions of symmetric cryptography and relations among them. Then we analyze security of authenticated encryption schemes designed by generic composition and introduce keyed hash functions and the NMAC scheme. The last topic we study is an authenticated encryption with associated data. Finally, we describe presented proofs generally and unify the methods used.

Keywords: symmetric cryptography, authenticated encryption, security proofs

Kapitola 1

Úvod

Kryptografické schéma je matematická metoda, která zajišťuje nějaký bezpečnostní cíl (integrita, utajení, ...). Zahrnuje celý proces zpracování dat a klíče, tedy všechny algoritmy, zobrazení, transformace a pravidla, která dané schéma využívá a podle kterých se řídí. Pro symetrická schémata platí, že pro šifrovací i dešifrovací algoritmus je použit stejný klíč. Značnou výhodou těchto schémat je pak nízká výpočetní náročnost.

Mluvíme-li o bezpečnosti kryptografických schémat, rozlišujeme několik typů. Zde uvedené důkazy se zabývají bezpečností dokazatelnou. Že je dané schéma bezpečné tedy znamená, že jeho prolomení je přibližně stejně obtížné jako řešení nějakého známého problému s velmi vysokou složitostí.

Cílem této práce bylo studium a následný jednotící pohled na metody těchto důkazů. Protože však oblast symetrické kryptografie je dosti rozsáhlá, práce je zaměřena především na studium článků týkajících se schémat šifrování s ochranou integrity. V každé kapitole je vždy představeno téma příslušného článku nebo jeho části a také výsledky, které přináší, včetně jednotlivých důkazů.

Ve druhé kapitole začínáme vymezením bezpečnostních pojmů symetrické kryptografie a dokazujeme, jaké vztahy mezi nimi platí. V navazující kapitole pak představíme generická schémata šifrování s ochranou integrity a ukážeme, jaké bezpečnostní vlastnosti jednotlivé metody jejich sestavování poskytují. Čtvrtá kapitola se pak zabývá využitím hašovacích funkcí k tvorbě MACu. Je zde také představeno schéma jedné z metod a je dokázána jeho bezpečnost. V předposlední kapitole pak ukážeme, jak lze klasické schéma šifrování s ochranou integrity přetvořit na schéma šifrování s ochranou integrity s připojenými daty. Uvedeme dva postupy a pro každý dokážeme, že takto vytvořené schéma je bezpečné. Závěrečná kapitola obsahuje shrnutí důkazové techniky a popis použitého postupu.

Kapitola 2

Bezpečnostní pojmy symetrického šifrování a vztahy mezi nimi

Tato i příští kapitola čerpá z článku [1]. Nejprve si představíme různé bezpečnostní pojmy pro symetrická šifrovací schémata a ukážeme, jaké vztahy mezi nimi platí. Tyto pojmy jsou definovány prostřednictvím her, na které se můžeme dívat jako na útoky na danou vlastnost šifrovacího schématu. Připomeňme pouze, že šifrovacím schématem $\mathcal{SE}$ rozumíme trojici $(\mathcal{K}, \mathcal{E}, \mathcal{D})$, což jsou po řadě algoritmus na tvorbu klíče, šifrovací a dešifrovací algoritmus.

2.1 Pojmy

Základními vlastnostmi schémat symetrického šifrování jsou *nerozlišitelnost* (indistinguishability) a *nepozměnitelnost* (non-malleability). Každou z těchto vlastností můžeme uvažovat buď při útoku s volbou otevřeného textu, nebo při útoku s volbou šifrového textu. Získáváme tak pojmy:

- nerozlišitelnost při útoku s volbou otevřeného textu (IND-CPA)
- nerozlišitelnost při útoku s volbou šifrového textu (IND-CCA)
- nepozměnitelnost při útoku s volbou otevřeného textu (NM-CPA)
- nepozměnitelnost při útoku s volbou šifrového textu (NM-CCA)

Další důležitou vlastností symetrických šifrovacích schémat je *integrita* (integrity). Rozlišujeme dva pojmy integrity:

- integrita otevřených textů (INT-PTXT)
- integrita šifrových textů (INT-CTXT)

2.2 Hry

Předpokládejme útočníka A. Na hru G se díváme jako na program. Skládá se z procedur (inicializační procedura INIT, procedury představující orákula a finální procedura FIN), které jsou spouštěny ve třech fázích. V první fázi je spuštěna procedura INIT. Ve druhé fázi je spuštěn útočník A, jehož vstupy jsou získány jako výstupy procedury INIT. A pokládá dotazy na orákula a jako odpovědi získává výstupy odpovídajících procedur hry G. Přitom platí, že odpověď na každý z dotazů je vytvářena nezávisle na ostatních. Třetí fáze nastává po ukončení běhu útočníka A. Spustí se procedura FIN a výstup této procedury je zároveň výstupem celé hry.

IND-CPA$_{\mathcal{SE}}$. V proceduře INIT první fáze hry se náhodně vybere klíč K z množiny všech klíčů $\mathcal{K}$ a bit b z množiny $\{0, 1\}$. Ve druhé fázi pak útočník A pokládá dotazy ve tvaru (M_0, M_1) (dvojice zpráv stejné délky) orákulu LR (pravolevé orákulum), které vrací jako odpověď zašifrovanou podobu zprávy M_b. Tato fáze končí, když útočník zvolí svůj odhad na hodnotu bitu b (tento odhad označujeme jako d). Do finální procedury třetí fáze pak vstupuje hodnota d, která je porovnána s hodnotou b. Výsledek tohoto srovnání je potom výsledkem celé hry. Shodují-li se obě hodnoty, znamená to, že útočník ve hře zvítězil (výstupem ze hry je hodnota true), v opačném případě prohrál (false).

Pro hodnotu advantage útočníka A platí:

$$\mathbf{Adv}^{\mathrm{ind-cpa}}_{\mathcal{SE}}(A) = 2 \cdot \Pr[\text{IND-CPA}^{A}_{\mathcal{SE}} \Rightarrow \text{true}] - 1.$$

IND-CCA$_{\mathcal{SE}}$. Tato hra probíhá podobně jako předchozí. V úvodní proceduře se navíc založí seznam S (jeho hodnotou je prázdná množina). Ve druhé fázi má útočník k dispozici kromě modifikovaného orákula LR, které si nyní ukládá odesílané odpovědi do seznamu S, ještě orákulum Dec. Jako dotazy na toto orákulum posílá útočník šifrové texty.

Orákulum nejprve ověří, zda příslušný šifrový text není v seznamu S (nebyl útočníkovi poslán jako odpověď orákula LR). Pokud není, vrátí jako odpověď příslušný otevřený text. Jinak vrací symbol $\perp$ pro neplatný text. Na konci této fáze útočník opět volí hodnotu d. Finální procedura již probíhá naprosto totožně s předchozí hrou.

Pro hodnotu advantage útočníka A platí:

$$\mathbf{Adv}^{\mathrm{ind-cca}}_{\mathcal{SE}}(A) = 2 \cdot \Pr[\text{IND-CCA}^{A}_{\mathcal{SE}} \Rightarrow \text{true}] - 1.$$

NM-CPA$_{\mathcal{SE}}$. Procedura INIT opět vybere náhodně klíč K a bit b a založí seznam S. Útočníkovi jsou nyní k dispozici orákula LR, Dec^* (dešifrovací orákulum) a Enc (šifrovací orákulum). Dotazy na orákulum LR jsou opět ve formě dvojic zpráv stejné délky. Orákulum nejprve ověří, zda útočník již neposlal dotaz na orákulum Dec^*. Pokud ano, vrací jako odpověď symbol $\perp$, pokud ne, vrací zašifrovanou podobu zprávy M_b a tuto odpověď uloží do seznamu S. Orákulum Dec^* smí být útočníkem dotazováno pouze jednou a po tomto dotazu je pak útočníkovi odepřen navíc i přístup k orákulu LR. Dotaz je položen ve formě vektoru C^* šifrových textů. Orákulum nejprve zaznamená, že dotaz na něj byl položen, a dále zpracovává jednotlivé komponenty vektoru C^*. Pro každý ověří, zda neleží v seznamu S a případně jej dešifruje. Odpovědí je pak vektor otevřených textů (nebo symbolů $\perp$, pokud příslušný $C^*[i]$ leží v S). Po tomto dotazu má útočník přístup již pouze k obyčejnému orákulu Enc, které dostává jako dotazy zprávy M a jako odpovědi posílá jim příslušné šifrové texty. Do konečné procedury FIN útočník opět posílá svůj odhad na bit b a pokud uhodne, ve hře vítězí.[1]

Pro hodnotu advantage útočníka A platí:

$$\mathbf{Adv}^{\mathrm{nm-cpa}}_{\mathcal{SE}}(A) = 2 \cdot \Pr[\text{NM-CPA}^{A}_{\mathcal{SE}} \Rightarrow \text{true}] - 1.$$

INT-PTXT$_{\mathcal{SE}}$. Inicializační procedura této hry náhodně vybere klíč K z množiny všech klíčů $\mathcal{K}$ a založí seznam S. Útočník má k dispozici šifrovací orákulum Enc a verifikační orákulum VF. Šifrovací orákulum dostává jako dotazy otevřené texty. Ty si ukládá do seznamu S a jako odpovědi vrací příslušné šifrové texty. Verifikačnímu orákulu jsou naopak posílány texty šifrové. Orákulum je dešifruje a ověřuje, zda odpovídající otevřený text je platný a pokud ano, zda neleží v seznamu S. Jsou-li

[1] Útok na nepozměnitelnost je zde definován jako paralelní útok na nerozlišitelnost s volbou šifrových textů.

obě podmínky splněny, znamená to, že útočník ve hře vyhrál (našel takový šifrový text, jenž mu nebyl zaslán jako odpověď šifrovacího orákula a zároveň se dešifruje na platný otevřený text). Odpovědí orákula je pak informace, zda otevřený text je platný, či nikoli. Finální procedura pak ohlásí, zda útočník zvítězil.

Pro hodnotu advantage útočníka A platí:

$$\mathbf{Adv}_{\mathcal{SE}}^{\mathrm{int-ptxt}}(A) = \Pr[\text{INT-PTXT}_{\mathcal{SE}}^{A} \Rightarrow \text{true}].$$

INT-CTXT$_{\mathcal{SE}}$. Tato hra probíhá víceméně totožně jako hra předchozí. Jediným rozdílem je, že do seznamu S jsou ukládány šifrové texty, které orákulum *Enc* vrací jako své odpovědi. Verifikační orákulum tedy nyní jako druhou podmínku ověřuje, zda dotaz jemu položený neleží v seznamu S.

Pro hodnotu advantage útočníka A platí:

$$\mathbf{Adv}_{\mathcal{SE}}^{\mathrm{int-ctxt}}(A) = \Pr[\text{INT-CTXT}_{\mathcal{SE}}^{A} \Rightarrow \text{true}].$$

2.3 Vztahy mezi pojmy

V prvním tvrzení ukážeme, že každé šifrovací schéma, které je bezpečné při útoku na integritu šifrových textů, je také bezpečné při útoku na integritu otevřených textů:

$$\text{INT-CTXT} \rightarrow \text{INT-PTXT}$$

Věta 1 *Mějme šifrovací schéma $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$. Potom pro libovolného útočníka A platí:*

$$\mathbf{Adv}_{\mathcal{SE}}^{\mathrm{int-ptxt}}(A) \leq \mathbf{Adv}_{\mathcal{SE}}^{\mathrm{int-ctxt}}(A).$$

Důkaz: Provedeme důkaz sporem. Předpokládejme, že útočník A vytvořil takový šifrový text C, že ve hře INT-PTXT$_{\mathcal{SE}}$ zvítězil. Potom označíme $M = \mathcal{D}(K, C)$. Dále nechť X je množina všech dotazů na šifrovací orákulum *Enc*, které útočník poslal předtím, než zvítězil, a Y je příslušná množina odpovědí na tyto dotazy.
Nyní ukážeme, že musí platit $M \notin X \Rightarrow C \notin Y$. Z toho již vyplývá, že když A zvítězil ve hře INT-PTXT$_{\mathcal{SE}}$ (poslal verifikačnímu orákulu takové C, jehož dešifrováním je otevřený text M, který není v X, tedy

nebyl poslán jako dotaz na šifrovací orákulum), pak zvítězil zároveň i ve hře INT-CTXT$_{\mathcal{SE}}$, protože C neleží v Y, tedy nebyl poslán jako odpověď šifrovacího orákula. To ale znamená, že advantage tohoto útočníka ve hře INT-CTXT$_{\mathcal{SE}}$ je menší nebo rovna jeho advantage ve hře INT-PTXT$_{\mathcal{SE}}$. Pro spor předpokládejme, že $C \in Y$. Potom ale musí existovat takové $x \in X$, že $C = \mathcal{E}(K, x)$. Protože dešifrování je jednoznačné, pak tedy musí také platit, že $x = \mathcal{D}(K, C) = M \in X$.

■

Další tvrzení nám říká, že pokud je šifrovací schéma bezpečné při útoku na nerozlišitelnost s volbou otevřených textů a zároveň je bezpečné i při útoku na integritu šifrových textů, je potom bezpečné také při útoku na nerozlišitelnost s volbou šifrových textů:

$$\text{INT-CTXT} \wedge \text{IND-CPA} \rightarrow \text{IND-CCA}$$

Věta 2 *Mějme šifrovací schéma $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$. Nechť A je útočník na nerozlišitelnost s volbou šifrových textů útočící na $\mathcal{SE}$, který běží v čase t a položí q_e dotazů orákulu LR a q_d dotazů dešifrovacímu orákulu Dec. Potom můžeme sestrojit útočníka na integritu šifrových textů A_c a útočníka na nerozlišitelnost s volbou otevřených textů A_p takové, že:*

$$\mathbf{Adv}^{\text{ind}-\text{cca}}_{\mathcal{SE}}(A) \leq 2 \cdot \mathbf{Adv}^{\text{int}-\text{ctxt}}_{\mathcal{SE}}(A_c) + \mathbf{Adv}^{\text{ind}-\text{cpa}}_{\mathcal{SE}}(A_p).$$

Dále platí, že útočník A_c běží v čase $O(t)$ a položí q_e dotazů na šifrovací orákulum Enc a q_d dotazů na verifikační orákulum VF, zatímco útočník A_p běží v čase $O(t)$ a položí q_e dotazů na orákulum LR.

Pro důkaz tohoto tvrzení budeme potřebovat následující nerovnost, která platí, pokud hry G_i a G_j probíhající identicky, dokud není booleovská proměnná *bad*, která se objevuje v obou těchto hrách, nastavena na hodnotu true[2]. Tato nerovnost vyplývá ze *Shoupova lemmatu*.

$$\Pr[G^A_i \Rightarrow y] - \Pr[G^A_j \Rightarrow y] \leq \Pr[G^A_j \text{ nastaví bad} \leftarrow \text{true}]. \qquad (2.1)$$

[2]Kód těchto her se liší pouze v příkazech, které následují po přenastavení této proměnné. Ta je na začátku hry explicitně nastavena na hodnotu $false$. Jakmile ale jednou získá hodnotu true, zůstává tato hodnota neměnná.

Důkaz (věty 2): Na začátku nejprve definujeme hry G_0, G_1, G_2.

Hra G_0: Inicializační procedura náhodně vybere klíč K a bit b a založí seznam S. Útočníkovi jsou pak k dispozici orákula LR a Dec. Dotazy na orákulum LR jsou opět ve tvaru dvojic zpráv stejné délky. Jako odpověď je odeslán šifrový text příslušný ke zprávě M_b, který je zároveň uložen do S. Orákulum Dec přijímá naopak šifrové texty. Nejprve ověří, zda se daný text nevyskytuje v seznamu S. Pokud ne, dešifruje ho jako zprávu M, pokud ano, prohlásí příslušnou zprávu M za neplatný text. Jedná-li se o platný otevřený text, nastaví booleovskou proměnnou *bad* na hodnotu true. Jako odpověď vrací M. V závěrečné proceduře útočník posílá svůj odhad na bit b.

Hra G_1: Hra probíhá identicky do momentu, než je přenastavena hodnota proměnné *bad*. Nyní ačkoliv útočník poslal jako dotaz na dešifrovací orákulum nový šifrový text, jehož dešifrováním vznikne platný otevřený text, orákulum mu odpoví symbolem $\perp$. V závěrečné proceduře útočník posílá svůj odhad na bit b.

Hra G_2: Úvodní procedura zvolí náhodně klíč K a bit b. K dispozici jsou opět zjednodušené pravolevé a dešifrovací orákulum. LR přijímá dvojice stejně dlouhých zpráv a odesílá jako odpověď šifrový text příslušný zprávě M_b. Dec na všechny dotazy odpovídá symbolem $\perp$. V závěrečné proceduře útočník posílá svůj odhad na bit b.

Nyní využijeme toho, že hra G_0 se útočníkovi A jeví úplně stejně jako hra IND-CCA. Můžeme tedy zapsat

$$\Pr[\text{IND-CCA}^A_{\mathcal{SE}} \Rightarrow \text{true}] = \Pr[G_0^A \Rightarrow \text{true}].$$

Výraz vpravo můžeme dále rozepsat následovně:

$$\begin{aligned}\Pr[\text{IND-CCA}^A_{\mathcal{SE}} \Rightarrow \text{true}] \;&=\; \Pr[G_1^A \Rightarrow \text{true}] + \\ &\quad +(\Pr[G_0^A \Rightarrow \text{true}] - \Pr[G_1^A \Rightarrow \text{true}]).\end{aligned}$$

Použijeme nerovnost (2.1), protože hry G_0, G_1 probíhají identicky, dokud není přenastavena hodnota proměnné *bad*. Dostáváme tak

$$\begin{aligned}\Pr[\text{IND-CCA}^A_{\mathcal{SE}} \Rightarrow \text{true}] \;&\leq\; \Pr[G_1^A \Rightarrow \text{true}] + \\ &\quad + \Pr[G_1^A \text{ nastaví bad} \leftarrow \text{true}]. \quad (2.2)\end{aligned}$$

Dále si všimněme her G_1 a G_2. Obě dešifrovací orákula zasílají jako odpovědi pouze symbol $\perp$, nezávisle na dotazovaném šifrovém textu. Útočníkovi se tedy opět jeví stejně a platí:

$$\Pr[G_1^A \Rightarrow \text{true}] = \Pr[G_2^A \Rightarrow \text{true}].$$

Nyní sestrojíme útočníky A_c a A_p:
Útočník A_c (ze znění věty víme, že útočí na integritu šifrových textů, tedy hraje hru INT-CTXT$_{\mathcal{SE}}$) na začátku svého běhu zvolí náhodný bit b. Poté spustí útočníka A, na jehož dotazy odpovídá pomocí vlastního orákula. Na dotaz tvaru (M_0, M_1) pro pravolevé orákulum vybere z dvojice zprávu M_b, tu pošle jako dotaz svému šifrovacímu orákulu a jeho odpověď vrátí útočníkovi A. Dotazy na dešifrovací orákulum pak přeposílá svému verifikačnímu orákulu, ale nezávisle na jeho odpovědi vrací útočníkovi A pouze symbol $\perp$.

Vidíme, že takto sestrojený útočník opravdu běží v čase $O(t)$ a položí q_e dotazů šifrovacímu orákulu a q_d dotazů verifikačnímu orákulu. Navíc platí následující nerovnost:

$$\Pr[G_1^A \text{ nastaví bad} \leftarrow \text{true}] \leq \Pr[\text{INT-CTXT}_{\mathcal{SE}}^{A_c} \Rightarrow \text{true}]. \tag{2.3}$$

Stačí si uvědomit, že hodnota proměnné *bad* se přenastaví pouze v případě, že útočník A nalezne takový šifrový text C, který se dešifruje na platný otevřený text, ale zároveň nebyl útočníkovi poskytnut šifrovacím orákulem. Stejná podmínka ale platí i pro vítězství ve hře INT-CTXT$_{\mathcal{SE}}$.

Útočník A_p (útočí na nerozlišitelnost s volbou otevřených textů, tedy hraje hru IND-CPA$_{\mathcal{SE}}$) spustí útočníka A. Jeho dotazy na pravolevé orákulum přeposílá svému pravolevému orákulu a odpovědi vrací, na dotazy na dešifrovací orákulum vždy odpoví $\perp$ a nepřeposílá je již nikam. I A_p běží v čase $O(t)$ a položí q_d dotazů orákulu LR. Navíc platí nerovnost:

$$\Pr[G_2^A \Rightarrow \text{true}] \leq \Pr[\text{IND-CPA}_{\mathcal{SE}}^{A_p} \Rightarrow \text{true}]. \tag{2.4}$$

A zvítězí ve hře G_2, pokud uhádne bit b. Tu samou hodnotu ale hádá i útočník A_p, takže kdykoli tuto hodnotu uhádne A, uhádne ji zároveň i A_p. Nerovnost (2.2) tedy můžeme s využitím (2.3) a (2.4) přepsat následovně:

$$\begin{aligned}\Pr[\text{IND-CCA}_{\mathcal{SE}}^{A} \Rightarrow \text{true}] \quad \leq \quad & \Pr[\text{INT-CTXT}_{\mathcal{SE}}^{A_c} \Rightarrow \text{true}] + \\ & + \Pr[\text{IND-CPA}_{\mathcal{SE}}^{A_p} \Rightarrow \text{true}].\end{aligned}$$

Přepíšeme-li celý výraz nyní podle definic advantage příslušných her, dostáváme:

$$\mathbf{Adv}^{\mathrm{ind-cca}}_{\mathcal{SE}}(A) \leq 2 \cdot \mathbf{Adv}^{\mathrm{ind-cpa}}_{\mathcal{SE}}(A_p) + \mathbf{Adv}^{\mathrm{int-ctxt}}_{\mathcal{SE}}(A_c),$$

čímž je důkaz u konce.

■

V následující větě ukážeme, že bezpečnost daného šifrovacího schématu při útoku na nerozlišitelnost s volbou šifrových textů nezaručuje bezpečnost při útoku na integritu otevřených textů:

IND-CCA $\nrightarrow$ INT-PTXT

Věta 3 *Nechť $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ je symetrické šifrovací schéma, které je bezpečné při útoku na nerozlišitelnost s volbou šifrových textů. Pak existuje symetrické šifrovací schéma $\mathcal{SE}_1$, které je také bezpečné při útoku na nerozlišitelnost s volbou šifrových textů, ale zároveň není bezpečné při útoku na integritu otevřených textů.*

Důkaz: Začneme tím, že definujeme schéma $\mathcal{SE}_1 = (\mathcal{K}, \mathcal{E}_1, \mathcal{D}_1)$ (pouze modifikujeme schéma $\mathcal{SE}$). Algoritmus pro generování klíče zůstává stejný jako ve schématu $\mathcal{SE}$. Šifrovací a dešifrovací algoritmy pak vypadají následovně:

Algoritmus $\mathcal{E}_1(K, M)$: Zprávu M zašifrujeme klíčem K pomocí šifrovací transformace schématu $\mathcal{SE}$ a získáme tak šifrový text C'. Výsledný šifrový text C pak dostaneme jako řetězec $0 \parallel C'$.

Algoritmus $\mathcal{D}_1(K, C)$: Ze vstupního šifrového texu oddělíme jeho první bit. Ten označíme jako b a zbylý řetězec jako C'. Pokud je hodnota bitu b rovna nule, dešifrujeme C' za použití klíče K pomocí dešifrovací transformace schématu $\mathcal{SE}$ a získáme tak zprávu M. V opačném případě položíme hodnotu zprávy M rovnu nule.

Nyní si ukážeme velmi jednoduchý útok na integritu šifrových textů schématu $\mathcal{SE}_1$, kde daný útočník A zvítězí s pravděpodobností jedna. Bude mu na to stačit jediný dotaz na verifikační orákulum VF, a to řetězec 10. Ten je pomocí transformace $\mathcal{D}_1$ dešifrován na otevřený text, jehož hodnota je 0 (tedy platný otevřený text). Na tento otevřený text

se ale útočník šifrovacího orákula nikdy nezeptal, tedy se nemohl objevit v seznamu S a podmínka pro vítězství je splněna. To nám říká, že $\mathbf{Adv}^{\mathrm{int-ptxt}}_{\mathcal{SE}_1}(A) = 1$, tedy schéma $\mathcal{SE}_1$ není bezpečné při útoku na integritu otevřených textů.

Abychom dokázali, že $\mathcal{SE}_1$ je bezpečné při útoku na nerozlišitelnost s volbou šifrových textů, ukážeme, že pro libovolného daného útočníka A, útočícího na $\mathcal{SE}_1$ v čase t a položivšího q_e dotazů na pravolevé orákulum LR a q_d dotazů na dešifrovací orákulum Dec, existuje útočník B, útočící na $\mathcal{SE}$ (to je dle předpokladu bezpečné při útoku na nerozlišitelnost s volbou šifrových textů) takový, že

$$\mathbf{Adv}^{\mathrm{ind-cca}}_{\mathcal{SE}_1}(A) \leq \mathbf{Adv}^{\mathrm{int-ptxt}}_{\mathcal{SE}}(B). \tag{2.5}$$

B poběží v čase $O(t)$ a položí q_e dotazů na pravolevé orákulum LR a q_d dotazů na dešifrovací orákulum Dec.

Útočník B je definován tak, že spustí útočníka A. Na jeho dotaz M_0, M_1 odpoví tak, že tuto dvojici předá svému pravolevému orákulu, k získané odpovědi připojí na začátek bit 0 a výsledný řetězec pošle zpět útočníkovi A. Při dotazu na dešifrovací orákulum pak nejprve rozdělí zaslaný šifrový text tak, že oddělí první bit. Je-li hodnota tohoto bitu rovna nule, pošle zbylý řetězec svému dešifrovacímu orákulu a získá hodnotu otevřeného textu. V opačném případě přiřadí otevřenému textu hodnotu nula. Výsledný otevřený text pak pošle zpět útočníkovi A. Po tom, co A ukončí svůj běh a pošle svůj odhad na bit b, vrátí útočník B tuto hodnotu.

Z tohoto popisu vidíme, že útočník B opravdu běží v čase $O(t)$ a položí stejné množství dotazů jako útočník A. Také je zřejmé, že kdykoli A uhodne bit b a zvítězí, uhodne tuto hodnotu i B a taktéž zvítězí. Nerovnost (2.5) tedy platí a dokázali jsme, že $\mathcal{SE}_1$ je bezpečné při útoku na nerozlišitelnost s volbou šifrových textů.

■

Poslední ze vztahů ukazuje, že bezpečnost šifrovacího schématu při útoku na integritu otevřených textů a bezpečnost při útoku na nerozlišitelnost s volbou otevřených textů nezaručuje nepozměnitelnost s volbou otevřených textů:

$$\text{INT-PTXT} \wedge \text{IND-CPA} \nrightarrow \text{NM-CPA}$$

Věta 4 *Nechť $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ je šifrovací schéma, které je bezpečné jak při útoku na integritu otevřených textů, tak při útoku na nerozlišitelnost*

s volbou otevřených textů. Pak existuje šifrovací schéma $\mathcal{SE}_2$, které je také bezpečné při útoku na integritu otevřených textů i při útoku na nerozlišitelnost s volbou otevřených textů, ale zároveň není bezpečné při útoku na nepozměnitelnost s volbou otevřených textů.

Důkaz: Při dokazování tohoto tvrzení opět nejprve zkonstruujeme schéma $\mathcal{SE}_2 = (\mathcal{K}, \mathcal{E}_2, \mathcal{D}_2)$ modifikací daného schématu $\mathcal{SE}$.

Algoritmus na generování klíče ponecháme totožný. Šifrovací algoritmus nejprve zašifruje vstupní otevřený text transformací $\mathcal{E}$ schématu $\mathcal{SE}$ za použití klíče K. Před takto získaný text pak připojí nulový bit a celý řetězec vrátí jako příslušný šifrový text. Dešifrování naopak probíhá tak, že se od vstupního šifrového textu oddělí první bit a zbylý řetězec se dešifruje transformací $\mathcal{D}$ schématu $\mathcal{SE}$ s klíčem K. Získaný otevřený text je pak již výsledný.

Nyní ukážeme útok na nepozměnitelnost s volbou otevřených textů schématu $\mathcal{SE}_2$, ve kterém útočník A zvítězí s pravděpodobností jedna. Útočník pošle pravolevému orákulu dvojici zpráv $0, 1$ a obdrží jako odpověď šifrový text C (ten je tvaru $0 \parallel C'$ a je takto i uložen v seznamu S). Nahradí první bit hodnotou 1 a tento řetězec uloží jako první složku vektoru C^* (tato je tvaru $1 \parallel C'$, takže neleží v S). C^* pak pošle orákulu Dec^*, které vrátí vektor P^*. Jako odhad bitu b je pak použita první složka tohoto vektoru. Před dešifrováním se totiž znovu oddělí první bit, takže do transformace vstupuje původní řetězec C'. Výsledkem je hodnota bitu b a útočník ve hře vítězí.
Ukázali jsme tak, že $\mathcal{SE}_2$ není bezpečné při útoku na nepozměnitelnost s volbou otevřených textů.

Dále ukážeme bezpečnost $\mathcal{SE}_2$ při útoku na integritu otevřených textů i při útoku na nerozlišitelnost s volbou otevřených textů. Stačí prokázat, že máme-li dány libovolné útočníky A_c a A_p na zmíněné vlastnosti schématu $\mathcal{SE}_2$, umíme sestrojit útočníky B_c a B_p útočící na $\mathcal{SE}$ takové, že

$$\mathbf{Adv}^{\mathrm{int-ptxt}}_{\mathcal{SE}_2}(A_c) \leq \mathbf{Adv}^{\mathrm{int-ptxt}}_{\mathcal{SE}}(B_c) \quad , \quad \mathbf{Adv}^{\mathrm{ind-cpa}}_{\mathcal{SE}_2}(A_p) \leq \mathbf{Adv}^{\mathrm{ind-cpa}}_{\mathcal{SE}}(B_p).$$

Navíc platí, že útočníci B_c a B_p běží v časech $O(A_c)$ a $O(A_p)$ a položí stejné počty dotazů.

Útočníky B_c a B_p sestrojíme následovně:
Útočník B_c nejprve spustí útočníka A_c. Na jeho dotazy na šifrovací orákulum odpovídá tak, že dotazovanou zprávu přepošle svému orákulu, před jeho odpověď připojí bit 0 a takto odešle zpět útočníkovi A_c. Při dotazu na verifikační orákulum naopak první bit vstupu odstraní, takto vzniklý řetězec pošle svému verifikačnímu orákulu a odpověď pošle útočníkovi A_c.

Vidíme, že pokud A_c ve hře zvítězí (pošle jako dotaz na VF takový šifrový text, že po odtržení prvního bitu se zbylý řetězec dešifruje na platný text, který zároveň neleží v seznamu S), znamená to, že zároveň zvítězil i útočník B_c.

Útočník B_p obdobně spustí útočníka A_p a na jeho dotazy na pravolevé orákulum odpovídá tak, že dvojici zpráv přepošle svému pravolevému orákulu a před získanou odpověď připojí bit 0. Tento řetězec pak vrátí útočníkovi A_p. Po ukončení běhu A_p odpoví odhad na bit b, který je zároveň odhadem útočníka B_p. Opět je zřejmé, že kdykoli A_p zvítězí, zvítězí i B_p, protože hádají stejnou hodnotu.

■

Kapitola 3

Generická schémata šifrování s ochranou integrity

V této části si představíme tři obecné příklady schémat šifrování s ochranou integrity[1]. Jedná se o jednoduché způsoby složení symetrického šifrovacího schématu se schématem autentizace zpráv. Tyto jednotlivé způsoby popíšeme a porovnáme jejich bezpečnost. Připomeňme pouze, že schéma autentizace zpráv $\mathcal{MA}$ je trojice $(\mathcal{K}, \mathcal{T}, \mathcal{V})$, což jsou po řadě algoritmus na generování klíče, tagovací a verifikační algoritmus.

Důležitým cílem schémat šifrování s ochranou integrity je *nefalzifikovatelnost* (unforgeability). Tuto vlastnost budeme uvažovat při útoku s volbou zpráv. Pro tato schémata tedy dostáváme ještě další dva pojmy bezpečnosti:

- slabá nefalzifikovatelnost při útoku s volbou zpráv (WUF-CMA)
- silná nefalzifikovatelnost při útoku s volbou zpráv (SUF-CMA)

3.1 Definice

Obecné kompozice. Předopokládáme, že máme dáno symetrické šifrovací schéma $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ a také schéma autentizace zpráv $\mathcal{MA} = (\mathcal{K}, \mathcal{T}, \mathcal{V})$. Dále předpokládáme, že dané šifrovací schéma je bezpečné při útoku na nerozlišitelnost s volbou otevřených textů, a dané schéma šifrování s ochranou integrity je bezpečné při útoku na nerozlišitelnost s volbou zpráv. Nyní představíme tři obecné metody jejich kompozice.

[1] Authenticated encryption schemes

***Encrypt-and-MAC* (E&M):** Otevřený text se pouze zašifruje a přidá se MAC[2] otevřeného textu. Dešifrování a ověření pak proběhne tak, že se MAC opět oddělí, zbylý řetězec se dešifruje a poté se ověří tag.

***MAC-than-encrypt* (MtE):** Nyní se nejprve vytvoří MAC otevřeného textu, ten se za něj připojí a takto vzniklý řetězec je teprve zaširován. Dešifrování a ověření proběhne tak, že celý řetězec se na začátku dešifruje, tím se získá dvojice (otevřený text, kandidát na tag), která se potom ověří.

***Encrypt-than-MAC* (EtM):** Při použití této metody se nejprve zašifruje otevřený text, ze získaného šifrového textu se udělá MAC, který je pak za šifrový text připojen. Dešifrování a ověření probíhá tak, že řetězec rozdělíme, nejprve ověříme tag a teprve poté dešifrujeme.

Představíme si ještě další dvě hry, pomocí nichž budeme dokazovat bezpečnost autentizačních šifrovacích chémat:

WUF-CMA$_{\mathcal{MA}}$. Inicializační procedura vybere náhodně klíč K a založí seznam S. Útočníkovi F jsou pak k dispozici tagovací a verifikační orákulum. Tagovacímu orákulu jsou zasílány zprávy M. Orákulum za použití klíče K vytvoří tag zprávy M, pošle ho zpět útočníkovi a dotazovanou zprávu si uloží do seznamu S. Verifikační orákulum získává jako vstup dvojici (M, τ). Nejprve za použití verifikačního algoritmu a klíče K ověří, zda zaslaný tag odpovídá zaslané zprávě, a výsledek stanoví jako hodnotu bitu b. Pokud je tato hodnota rovna jedné a zároveň platí, že dotazovaná zpráva neleží v seznamu S, znamená to, že útočník ve hře zvítězil.

Pro hodnotu advantage útočníka F platí:

$$\mathbf{Adv}^{\mathrm{wuf-cma}}_{\mathcal{MA}}(F) = \Pr[\text{WUF-CMA}^{F}_{\mathcal{MA}} \Rightarrow \text{true}].$$

SUF-CMA$_{\mathcal{MA}}$. Opět je náhodně zvolen klíč K a založen seznam S a útočník F má k dispozi tagovací a verifikační orákula, která pracují velice podobně, jako v předchozí hře. Činnost tagovacího orákula se od předešlého případu liší pouze v tom, že si do seznamu S zaznamenává celou dvojici (M, τ). Verifikační orákulum pak tedy po úspěšné verifikaci

[2]MAC (message authentication code) je funkce, která má na vstupu tajný klíč a zprávu libovolné délky a na výstupu řetězec pevné délky (tag, někdy nazývaný MAC). Tato hodnota pak umožňuje ověřit autenticitu a integritu posílané zprávy.

ověřuje, zda celá dotazovaná dvojice neleží v seznamu S. Pokud je tato podmínka splněna, útočník ve hře vítězí.

Pro hodnotu advantage útočníka F platí:

$$\mathbf{Adv}^{\mathrm{suf-cma}}_{\mathcal{MA}}(F) = \Pr[\text{SUF-CMA}^{F}_{\mathcal{MA}} \Rightarrow \text{true}].$$

3.2 Bezpečnost generických schémat

V následujícím oddíle podrobněji popíšeme jednotlivé metody kombinování schémat a podíváme se, jaké typy bezpečnosti zaručují. Budeme také předpokládat, že máme dané symetrické šifrovací schéma $\mathcal{SE} = (\mathcal{K}_e, \mathcal{E}, \mathcal{D})$, které je bezpečné při útoku na nerozlišitelnost s volbou otevřených textů (IND-CPA bezpečné), a dále že máme dané schéma autentizace zpráv $\mathcal{MA} = (\mathcal{K}_m, \mathcal{T}, \mathcal{V})$, které je buď slabě nebo silně nefalzifikovatelné při útoku s volbou zpráv. Konkrétní kombinované schéma budeme značit $\overline{\mathcal{SE}} = (\overline{\mathcal{K}}, \overline{\mathcal{E}}, \overline{\mathcal{D}})$.

3.2.1 Encrypt-and-MAC

Mějme dána schémata $\mathcal{SE} = (\mathcal{K}_e, \mathcal{E}, \mathcal{D})$ a $\mathcal{MA} = (\mathcal{K}_m, \mathcal{T}, \mathcal{V})$. Jejich zkombinováním metodou **E&M** získáme schéma $\overline{\mathcal{SE}} = (\overline{\mathcal{K}}, \overline{\mathcal{E}}, \overline{\mathcal{D}})$, kde jednotlivé algoritmy jsou následující:

Algoritmus $\overline{\mathcal{K}}$: Nejprve náhodně vybereme klíč K_e z množiny všech klíčů šifrovacího schématu $\mathcal{SE}$, potom náhodně vybereme klíč K_m z množiny všech klíčů autentizačního schématu $\mathcal{MA}$ a jako hodnotu klíče $\overline{K}$ pak vezmeme zřetězení $K_e \parallel K_m$.

Algoritmus $\overline{\mathcal{E}} = (K_e \parallel K_m, M)$: Na začátku zašifrujeme zprávu M šifrovacím algoritmem $\mathcal{E}$ schématu $\mathcal{SE}$ pomocí klíče K_e (takto získaný šifrový text označíme C') a také vytvoříme její tag pomocí tagovacího algoritmu $\mathcal{T}$ schématu $\mathcal{MA}$ za použití klíče K_m (tag označíme symbolem τ). Výsledný šifrový text pak vznikne zřetězením $C' \parallel \tau$.

Algoritmus $\overline{\mathcal{D}} = (K_e \parallel K_m, C)$: Vstupní šifrový text C nejprve rozdělíme na původní šifrový text C' a tag τ. C' dešifrujeme algoritmem $\mathcal{D}$ schématu $\mathcal{SE}$ pomocí klíče K_e a získáme tak zprávu M. Tuto zprávu pak

společně s tagem τ ověříme verifikačním algoritmem $\mathcal{V}$ schématu $\mathcal{MA}$ za použití klíče K_m. Proběhne-li ověření úspěšně, vrátí algoritmus jako odpověd hodnotu zprávy M, v opačném případě symbol $\perp$.

Nyní ukážeme, že takto vytvořené schéma poskytuje integritu otevřených textů. Tato vlastnost je přímým důsledkem integrity MACu, tedy nezávisí na daném symetrickém šifrovacím schématu.

Věta 5 *Nechť $\mathcal{SE} = (\mathcal{K}_e, \mathcal{E}, \mathcal{D})$ je symetrické šifrovací schéma, $\mathcal{MA} = (\mathcal{K}_m, \mathcal{T}, \mathcal{V})$ nechť je schéma autentizace zpráv a ať $\overline{\mathcal{SE}} = (\overline{\mathcal{K}}, \overline{\mathcal{E}}, \overline{\mathcal{D}})$ je kombinované schéma získané pomocí metody Encrypt-and-MAC. Pak pro jakéhokoli daného útočníka A, útočícího na $\overline{\mathcal{SE}}$, můžeme sestrojit útočníka F tak, že*

$$\mathbf{Adv}^{\mathrm{int-ptxt}}_{\overline{\mathcal{SE}}}(A) \leq \mathbf{Adv}^{\mathrm{wuf-cma}}_{\mathcal{MA}}(F).$$

Důkaz: Sestrojíme útočníka F využívajícího ve svém běhu útočníka A a ukážeme, že podaří-li se útočníkovi A najít takový šifrový text C, jehož příslušný otevřený text je platný a nebyl předem zaslán jako dotaz na šifrovací orákulum, znamená to, že se mu také podařilo najít dvojici (zpráva, tag) pro útočníka F, která bude ověřena jako platná. Z toho pak již přímo vyplývá dokazované tvrzení.

Útočník F bude pracovat následovně: Na začátku běhu zvolí náhodně klíč K_e z množiny všech klíčů $\mathcal{K}_e$ a poté spustí útočníka A. Jeho dotazy na šifrovací orákulum zodpovídá tak, že příslušnou zprávu nejprve přepošle svému tagovacímu orákulu, dále tuto zprávu zašifruje transformací $\mathcal{E}$ schématu $\mathcal{SE}$ za použití klíče K_e, za takto získaný šifrový text připojí tag a celé vrátí útočníkovi A. Ptá-li se útočník A verifikačního orákula, útočník F naopak nejprve dotaz rozdělí na šifrový text a tag. Šifrový text dešifruje pomocí dešifrovacího algoritmu schématu $\mathcal{SE}$ a získaný otevřený text spolu s tagem zašle svému verifikačnímu orákulu. Jeho odpověď pak vrátí útočníkovi A.

Vidíme, že útočník A zvítězí, pokud jím zaslaný šifrový text je tvaru $C = C' \parallel \tau$, kde $M = \mathcal{D}(K_e, C')$ je platný šifrový text, který neleží v seznamu dotazů na šifrovací orákulum, a zároveň pokud verifikační algoritmus schématu $\mathcal{MA}$ ověří dvojici (M, τ) jako platnou. Tato dvojice je ale přesně tou, kterou hledá útočník F (ověření se podaří a M neleží v seznamu dotazů na tagovací orákulum). Advantage útočníka F je tedy minimálně stejně velká jako advantage útočníka A a důkaz je u konce.

■

E&M neposkytuje IND-CPA, IND-CCA a NM-CPA bezpečnost. Dál si ukážeme, že toto schéma může poskytnout útočníkovi dodatečné informace o otevřeném textu, a to takové, že bude schopen úspěšně útočit na nerozlišitelnost s volbou otevřených textů.

Útočníkovi stačí pouze na začátku útoku zvolit dvě stejně dlouhé, ale různé zprávy x, y. Prvním dotazem na pravolevé orákulum je právě tato dvojice. Jako odpověď obdrží útočník šifrový text tvaru $C'_1 \parallel \tau_1$. Poté pošle dotaz (x, x) a obdrží jako odpověď šifrový text tvaru $C'_2 \parallel \tau_2$. Nyní již stačí pouze porovnat obě hodnoty tagů. Shodují-li se, jeho odhad bitu b bude 0, v opačném případě 1. Je zřejmé, že tento útočník zvítězí s pravděpodobností jedna, a tedy schéma není IND-CPA bezpečné.

Protože platí, že je-li schéma IND-CCA a zároveň NM-CPA bezpečné, pak je i IND-CPA bezpečné, neposkytuje schéma E&M ani jednu z těchto bezpečností.

E&M neposkytuje INT-CTXT bezpečnost. K prokázání tohoto tvrzení si stačí uvědomit, že existují taková šifrovací schémata, která jsou IND-CPA bezpečná a zároveň v nich platí, že šifrový text může být pozměněn bez toho, aby se změnil otevřený text, který se získá jeho dešifrováním. Pokud je takovéto šifrovací schéma použito v kompilaci, může útočník jednoduše zaslat požadavek na šifrovací orákulum, získaný šifrový text vhodným způsobem pozměnit a zaslat jej takto verifikačnímu orákulu. Tento text jistě neleží v seznamu odpovědí šifrovacího orákula, tedy verifikací projde a útočník zvítězí.

Jako příklad poslouží již popsané schéma $\mathcal{SE}_2$, odvozené ze schématu $\mathcal{SE}$, o kterém předpokládáme, že je IND-CPA bezpečné. Připomeňme, že šifrovací algoritmus $\mathcal{E}_2$ zašifruje vstupní zprávu pomocí algoritmu $\mathcal{E}$ a předsadí před takto vzniklý text nulový bit. Dešifrovací algoritmus $\mathcal{D}_2$ pak naopak nejdříve odstraní první bit vstupu a poté získaný text dešifruje pomocí algoritmu $\mathcal{D}$. Pro toto schéma jsme ukázali, že je také IND-CPA bezpečné. Nyní už stačí pouze sestrojit útočníka A, útočícího na INT-CTXT vlastnost schématu $\overline{\mathcal{SE}}$, a ukázat, že jeho advantage je rovna jedné.

Útočníkovi A stačí, aby se šifrovacího orákula zeptal na zprávu M. Obdrží pak odpověď tvaru $0 \parallel \overline{\mathcal{E}}(\overline{K}, M)$. Tento text modifikuje tak, že změní hodnotu prvního bitu (získá $1 \parallel \overline{\mathcal{E}}(\overline{K}, M)$), a pošle jej verifikačnímu orákulu. Tento text neleží v seznamu dotazů a dešifruje se na původní zprávu M, tedy útočník vítězí.

3.2.2 MAC-than-Encrypt

Opět máme dána schémata $\mathcal{SE} = (\mathcal{K}_e, \mathcal{E}, \mathcal{D})$ a $\mathcal{MA} = (\mathcal{K}_m, \mathcal{T}, \mathcal{V})$. Jejich zkombinováním metodou **MtE** získáme schéma $\overline{\mathcal{SE}} = (\overline{\mathcal{K}}, \overline{\mathcal{E}}, \overline{\mathcal{D}})$, kde jednotlivé algoritmy jsou následující:

Algoritmus $\overline{\mathcal{K}}$: Náhodně vybereme klíč K_e z množiny $\mathcal{K}_e$, náhodně vybereme klíč K_m z množiny $\mathcal{K}_m$ a jako hodnotu klíče $\overline{K}$ pak vezmeme zřetězení $K_e \parallel K_m$.

Algoritmus $\overline{\mathcal{E}} = (K_e \parallel K_m, M)$: Na začátku vytvoříme tag τ zprávy M pomocí tagovacího algoritmu $\mathcal{T}$ s klíčem K_m, připojíme ho k této zprávě a celé zašifrujeme šifrovacím algoritmem $\mathcal{E}$ pomocí klíče K_e.

Algoritmus $\overline{\mathcal{D}} = (K_e \parallel K_m, C)$: Vstupní šifrový text C nejprve dešifrujeme algoritmem $\mathcal{D}$ za použití klíče K_e. Vzniklý text rozdělíme na zprávu M a tag τ. Tuto zprávu pak společně s tagem τ ověříme verifikačním algoritmem $\mathcal{V}$ za použití klíče K_m. Proběhne-li ověření úspěšně, vrátí algoritmus jako odpověd hodnotu zprávy M, v opačném případě symbol $\perp$.

Pro schéma získané touto metodou dokážeme, že poskytuje integritu otevřených textů a také je bezpečné při útoku na nerozlišitelnost s volbou otevřených textů. Integrita je zaručena díky WUF-CMA vlastnosti schématu $\mathcal{MA}$ a IND-CPA vlastnost se opět dědí od šifrovacího schématu $\mathcal{SE}$.

Věta 6 *Nechť $\mathcal{SE} = (\mathcal{K}_e, \mathcal{E}, \mathcal{D})$ je symetrické šifrovací schéma, $\mathcal{MA} = (\mathcal{K}_m, \mathcal{T}, \mathcal{V})$ nechť je schéma autentizace zpráv a ať $\overline{\mathcal{SE}} = (\overline{\mathcal{K}}, \overline{\mathcal{E}}, \overline{\mathcal{D}})$ je kombinované schéma získané pomocí metody MAC-than-Encrypt. Pak pro jakéhokoli daného útočníka I útočícího na $\overline{\mathcal{SE}}$ můžeme sestrojit útočníka F tak, že:*

$$\mathbf{Adv}^{\mathrm{int-ptxt}}_{\overline{\mathcal{SE}}}(I) \leq \mathbf{Adv}^{\mathrm{wuf-cma}}_{\mathcal{MA}}(F).$$

Důkaz: Opět budeme postupovat tak, že sestrojíme útočníka F využívajícího daného útočníka I a ukážeme, že kdykoli se podaří útočníkovi I zvítězit, znamená to vítězství i pro útočníka F.

Útočník F (útočí na WUF-CMA vlastnost schématu $\mathcal{MA}$) tedy nejprve náhodně zvolí klíč K_e z množiny $\mathcal{K}_e$ a následně spustí útočníka I. Ten útočí na integritu otevřených textů schématu $\overline{\mathcal{SE}}$, tedy se snaží najít takový šifrový text C, jehož dešifrováním algoritmem $\overline{\mathcal{D}}$ je získán platný otevřený text, jež nebyl útočníkem zaslán jako dotaz na šifrovací orákulum. Pošle-li I dotaz na šifrovací orákulum, F tento dotaz přepošle svému tagovacímu orákulu, vrácený tag připojí za původní zprávu, celé zašifruje algoritmem $\mathcal{E}$ s klíčem K_e a získaný šifrový text vrátí útočníkovi I. Při dotazu na verifikační orákulum útočníka I útočník F nejprve dotazovaný šifrový text dešifruje algoritmem $\mathcal{D}$ s klíčem K_e, získá tak text tvaru $M \parallel \tau$ a dvojici (M, τ) zašle k ověření svému verifikačnímu orákulu. Odpověď orákula pak vrátí útočníkovi I.

Vidíme, že pokud I našel C takové, že $\mathcal{D}(K_e, C) = M \parallel \tau$, kde M je platný text, který neleží v seznamu dotazů na šifrovací orákulum, a zároveň že verifikace $\mathcal{V}(K_m, M, \tau)$ proběhne úspěšně, znamená to, že našel vítěznou dvojici (M, τ) pro útočníka F. Jeho advantage je tedy stejná nebo vyšší a tvrzení platí.

■

Věta 7 *Nechť $\mathcal{SE} = (\mathcal{K}_e, \mathcal{E}, \mathcal{D})$ je symetrické šifrovací schéma, $\mathcal{MA} = (\mathcal{K}_m, \mathcal{T}, \mathcal{V})$ nechť je schéma autentizace zpráv a ať $\overline{\mathcal{SE}} = (\overline{\mathcal{K}}, \overline{\mathcal{E}}, \overline{\mathcal{D}})$ je kombinované schéma získané pomocí metody MAC-than-Encrypt. Pak pro jakéhokoli daného útočníka A, útočícího na $\overline{\mathcal{SE}}$, můžeme sestrojit útočníka A_p tak, že:*

$$\mathbf{Adv}^{\mathrm{ind-cpa}}_{\overline{\mathcal{SE}}}(A) \leq \mathbf{Adv}^{\mathrm{ind-cpa}}_{\mathcal{MA}}(A_p).$$

Důkaz: Stejně jako v předchozím důkazu sestrojíme útočníka A_p útočícího na IND-CPA vlastnost schématu $\mathcal{SE}$ s využitím daného útočníka A útočícího na tutéž vlastnost schématu $\overline{\mathcal{SE}}$.

A_p náhodně zvolí klíč K_m z množiny $\mathcal{K}_m$ a spustí útočníka A. Ten posílá dotazy na pravolevé orákulum ve tvaru (M_0, M_1). Útočník A obě tyto zprávy doplní o jejich tagy (získá je použitím tagovacího algoritmu $\mathcal{T}$ a klíče K_m) a tuto novou dvojici pošle svému pravolevému orákulu. Odpověď pak přímo pošle útočníkovi A_p. Na konci svého běhu A posílá odhad na hodnotu bitu b. Tento odhad použije i A_p.

Je zřejmé, že pokud A uhádl a zvítězil, to samé musí platit i pro A_p, protože hádají stejnou hodnotu bitu b. Z toho již dokazovaná nerovnost bezprostředně vyplývá.

■

MtE neposkytuje NM-CPA, IND-CCA ani INT-CTXT bezpečnost. Předpokládejme opět, že šifrovací schéma použité v kombinovaném schématu $\overline{\mathcal{SE}}$ metodou *MAC-than-Encrypt* je schéma $\mathcal{SE}_2$. Ukážeme útočníka A, který zvítězí při útoku na NM-CPA vlastnost schématu $\overline{\mathcal{SE}}$ s pravděpodobností jedna. Takto prokáže, že toto schéma není NM-CPA bezpečné, z čehož přímo vyplývá, že není ani IND-CCA bezpečné (je-li IND-CCA, je zároveň NM-CCA a tedy i NM-CPA). Zároveň proto nemůže být ani INT-CTXT bezpečné (je-li IND-CPA i INT-CTXT, pak musí být IND-CCA).

Útočník A pošle pravolevému orákulu dotaz tvaru $(0, 1)$. Orákulum vezme zprávu M_b, vytvoří její tag τ pomocí tagovacího algoritmu $\mathcal{T}$ a klíče K_m, připojí ho za M_b, celý řetězec zašifruje algoritmem $\mathcal{E}_2$ s klíčem K_e, pošle výsledný text (tedy $C = 0 \parallel \mathcal{E}(K_e, M_b \parallel \tau)$) jako odpověď útočníkovi A a uloží si ho do seznamu S. Útočník změní hodnotu prvního bitu, vzniklý text položí jako první složku vektoru C^* a tento vektor pošle dešifrovacímu orákulu Dec^*. Toto orákulum při zpracování první složky dotazu nejprve ověří, zda se nenachází v seznamu S. Potom teprve text dešifruje tak, že odebere první bit (získá tak $C' = \mathcal{E}(K_e, M_b \parallel \tau)$), zbylý řetězec dešifruje algoritmem $\mathcal{D}$ s klíčem K_e, získá tak text $M' = M_b \parallel \tau$, dvojici ověří a pošle útočníkovi zprávu M_b. To je ale hodnota bitu b.

3.2.3 Encrypt-than-MAC

Schéma $\overline{\mathcal{SE}} = (\overline{\mathcal{K}}, \overline{\mathcal{E}}, \overline{\mathcal{D}})$ získáme metodou **EtM** z daných schémat $\mathcal{SE} = (\mathcal{K}_e, \mathcal{E}, \mathcal{D})$ a $\mathcal{MA} = (\mathcal{K}_m, \mathcal{T}, \mathcal{V})$. Jednotlivé algoritmy schématu $\overline{\mathcal{SE}}$ jsou následující:

Algoritmus $\overline{\mathcal{K}}$: Náhodně vybereme klíč K_e z množiny $\mathcal{K}_e$, náhodně vybereme klíč K_m z množiny $\mathcal{K}_m$ a jako hodnotu klíče $\overline{K}$ pak vezmeme zřetězení $K_e \parallel K_m$.

Algoritmus $\overline{\mathcal{E}} = (K_e \parallel K_m, M)$: Nejprve zašifrujeme zprávu M algoritmem $\mathcal{E}$ s klíčem K_e. Poté vytvoříme tag τ takto vzniklého šifrového textu C' pomocí algoritmu $\mathcal{T}$ a klíče K_m. Výsledným textem pak bude zřetězení $C' \parallel \tau$.

Algoritmus $\overline{\mathcal{D}} = (K_e \parallel K_m, C)$: Vstupní text C nejprve rozdělíme na šifrový text C' a tag τ. C' dešifrujeme algoritmem $\mathcal{D}$ za použití klíče K_e a získáme tak zprávu M. Dále ověříme dvojici (C', τ) verifikačním algoritmem $\mathcal{V}$ za použití klíče K_m. Proběhne-li ověření úspěšně, vrátí algoritmus jako odpověď hodnotu zprávy M, v opačném případě symbol $\perp$.

V tomto oddíle již dosažené výsledky závisí na tom, jak je bezpečné schéma autentizace zpráv použité v kompilaci. Nejprve se tedy budeme zabývat schématy s MACem, který je WUF-CMA bezpečný, a poté ukážeme výsledky pro schémata, jejichž MAC je SUF-CMA bezpečný.

Následující tvrzení říkají, že máme-li kombinované schéma, jehož MAC je WUF-CMA bezpečný, toto schéma poskytuje IND-CPA a INT-PTXT bezpečnost.

Věta 8 *Nechť $\mathcal{SE} = (\mathcal{K}_e, \mathcal{E}, \mathcal{D})$ je symetrické šifrovací schéma, $\mathcal{MA} = (\mathcal{K}_m, \mathcal{T}, \mathcal{V})$ nechť je schéma autentizace zpráv a ať $\overline{\mathcal{SE}} = (\overline{\mathcal{K}}, \overline{\mathcal{E}}, \overline{\mathcal{D}})$ je kombinované schéma získané pomocí metody Encrypt-than-MAC. Pak pro jakéhokoli daného útočníka A, útočícího na $\overline{\mathcal{SE}}$, můžeme sestrojit útočníka A_p tak, že:*

$$\mathbf{Adv}^{\mathrm{ind-cpa}}_{\overline{\mathcal{SE}}}(A) \leq \mathbf{Adv}^{\mathrm{ind-cpa}}_{\mathcal{MA}}(A_p).$$

Důkaz: Sestrojíme útočníka A_p, využívajícího daného útočníka A následujícím způsobem: Útočník nejprve náhodně vybere klíč K_m z množiny klíčů $\mathcal{K}_m$ a poté spustí útočníka A. Jeho dotazy na pravolevé orákulum útočník A_p přeposílá svému pravolevému orákulu. To mu vrací šifrový text C. Útočník za použití klíče K_m vytvoří tag τ tohoto textu a jako odpověď útočníkovi A pošle zřetězení $C \parallel \tau$. Na konci svého běhu útočník A pošle svůj odhad na bit b. Tento odhad potom použije i útočník A_p.

Vidíme, že kdykoli uhodne A hodnotu bitu b správně, uhodne ji i A_p, tedy požadovaná nerovnost opravdu platí.

■

Věta 9 *Nechť $\mathcal{SE} = (\mathcal{K}_e, \mathcal{E}, \mathcal{D})$ je symetrické šifrovací schéma, $\mathcal{MA} = (\mathcal{K}_m, \mathcal{T}, \mathcal{V})$ nechť je schéma autentizace zpráv a ať $\overline{\mathcal{SE}} = (\overline{\mathcal{K}}, \overline{\mathcal{E}}, \overline{\mathcal{D}})$ je kombinované schéma získané pomocí metody Encrypt-than-MAC. Pak pro jakéhokoli daného útočníka I, útočícího na $\overline{\mathcal{SE}}$, můžeme sestrojit útočníka F tak, že:*

$$\mathbf{Adv}^{\mathrm{int-ptxt}}_{\overline{\mathcal{SE}}}(I) \leq \mathbf{Adv}^{\mathrm{wuf-cma}}_{\mathcal{MA}}(F).$$

Důkaz: Tvrzení opět dokážeme tak, že sestrojíme útočníka F útočícího na WUF-CMA vlastnost schématu $\mathcal{MA}$ (snaží se najít dvojici (M, τ), kde zprávu M předem nepoužil jako dotaz na šifrovací orákulum, takovou, že verifikace proběhne v pořádku) jež ve svém běhu využívá daného útočníka I. Ten útočí na INT-PTXT vlastnost schématu $\overline{\mathcal{SE}}$, tedy se snaží pomocí dotazů na šifrovací orákulum najít takový šifrový text, který bude přijat verifikačním orákulem jako platný.

F náhodně vybere klíč K_e z množiny klíčů $\mathcal{K}_e$ a spustí útočníka I. Na jeho dotazy na šifrovací orákulum odpovídá tak, že dotazovanou zprávu M zašifruje pomocí klíče K_e. Získaný šifrový text C' pošle svému tagovacímu orákulu a obdrží tak tag τ. Jako odpověď útočníkovi I pak pošle zřetězení $C' \parallel \tau$. Při dotazu na verifikační orákulum pak F dotazovaný šifrový text rozdělí na $C' \parallel \tau'$, svému verifikačnímu orákulu pošle dotaz tvaru (C', τ') a jeho odpověď přepošle útočníkovi I.

Nyní ukážeme, že kdykoli I porazí INT-PTXT vlastnost schématu $\overline{\mathcal{SE}}$, pak nutně F porazil WUF-CMA vlastnost schématu $\mathcal{MA}$. Předpokládejme, že I našel takový šifrový text $C = C' \parallel \tau'$, jež jeho verifikační orákulum ověří jako platný. To znamená, že $\mathcal{D}(K_e, C') = M$ je platný otevřený text, který nebyl útočníkem předem zaslán jako dotaz na jeho šifrovací orákulum, a $\mathcal{V}(K_m, C', \tau')$ vrátí hodnotu 1. Ovšem díky jednoznačnosti dešifrování také platí, že šifrový text C' příslušný zprávě M nemohl být zaslán útočníkem F jako dotaz na tagovací orákulum, tedy i v jeho případě verifikace $\mathcal{V}(K_m, C', \tau')$ proběhne v pořádku.

■

Dále tvrdíme, že schéma získané metodou *Encrypt-than-MAC*, jehož schéma šifrování s ochranou integrity je slabě nefalzifikovatelné, nezajišťuje bezpečnost při útoku na nepozměnitelnost s volbou otevřených textů a nerozlišitelnost s volbou šifrových textů. Také nezajišťuje integritu šifrových textů.

EtM s WUF-CMA bezpečným MACem neposkytuje NM-CPA bezpečnost. Předpokládejme, že máme dáno schéma autentizace zpráv $\mathcal{MA}= (\mathcal{K}, \mathcal{T}, \mathcal{V})$, které je WUF-CMA bezpečné. Nyní vytvoříme schéma $\mathcal{MA}'= (\mathcal{K}, \mathcal{T}, \mathcal{V})$ ze schématu $\mathcal{MA}$ takové, že kombinované schéma $\overline{\mathcal{SE}}$ vytvořené metodou *Encrypt-than-MAC* ze schémat $\mathcal{SE}$ a $\mathcal{MA}'$ nebude NM-CPA bezpečné.

Schématu $\mathcal{MA}'= (\mathcal{K}, \mathcal{T}', \mathcal{V}')$ ponecháme původní algoritmus na generování klíče $\mathcal{K}$ a definujeme tagovací algoritmus a verifikační následovně:

Algoritmus $\mathcal{T}'(K, M)$**:** Nejprve je získán tag τ použitím algoritmu $\mathcal{T}$, za nějž se pak připojí nulový bit ($\tau \parallel 0$).

Algoritmus $\mathcal{V}'(K, M, \tau)$**:** Z tagu τ je nejprve odstraněn poslední bit a zbylý řetězec se ověří pomocí algoritmu $\mathcal{V}$.

Dále sestrojíme takového útočníka A, který s pravděpodobností jedna porazí NM-CPA vlastnost schématu $\overline{\mathcal{SE}}$. Útočník nejprve pošle dotaz na pravolevé orákulum tvaru $(0, 1)$ a obdrží šifrový text $C = C' \parallel \tau'$, kde $C' = \mathcal{E}(K_e, M_b)$ a $\tau' = \mathcal{T}(K_m, C')$. Poslední bit tagu τ' změní na hodnotu 1 a získá tak tag τ. Jako první složku vektoru $C*$ pak použije zřetězení $C' \parallel \tau$. Vektor $C*$ pošle dešifrovacímu orákulu. První složka vektoru odpovědí $P*$ je pak zřejmě hodnoutou bitu b a útočník jistě vítězí. Dokázali jsme tedy, že kombinované schéma $\overline{\mathcal{SE}}$ není NM-CPA bezpečné.

Nyní ještě zbývá důkaz WUF-CMA bezpečnosti schématu $\mathcal{MA}'$. Předpokládejme, že máme zadaného libovolného útočníka A, útočícího na WUF-CMA vlastnost schématu $\mathcal{MA}'$. Sestrojíme útočníka A_w, který bude útočit na WUF-CMA vlastnost schématu $\mathcal{MA}$ a bude při útoku využívat útočníka A. Útočník A_w pracuje jednoduše tak, že pouze spustí útočníka A. Na jeho dotazy na tagovací orákulum odpovídá tak, že dotaz přepošle svému tagovacímu orákulu, za jeho odpověď připojí nulový bit a vrátí takto útočníkovi A. Při dotazu útočníka A na verifikační orákulum tvaru (M, τ) A_w nejprve odstraní poslední bit tagu τ a získá tak tag τ'. Svému verifikačnímu orákulu pak pošle dotaz tvaru (M, τ') a jeho odpověď vrátí útočníkovi A. Vidíme, že pokud útočník A našel takovou dvojici (M, τ), že ověření $\mathcal{V}'(K, M, \tau)$ proběhne v pořádku, znamená to, že proběhlo v pořádku i ověření $\mathcal{V}(K, M, \tau')$ a tedy kdykoli zvítězí útočník A, jistě zvítězí i A_w. Protože jsme ale předpokládali, že schéma $\mathcal{MA}$ je WUF-CMA bezpečné, musí totéž platit i pro schéma $\mathcal{MA}'$.

EtM s WUF-CMA bezpečným MACem neposkytuje IND-CCA ani INT-CTXT bezpečnost. Na začátku této sekce jsme ukázali, že schéma vytvořené metodou *Encrypt-than-MAC*, jehož schéma autentizace zpráv je slabě nefalzifikovatelné, zajišťuje IND-CPA bezpečnost. Dále víme, že je-li schéma zároveň INT-CTXT a IND-CPA bezpečné, pak je i NM-CPA bezpečné, což je spor s předchozím tvrzením. Podobně

platí, že je-li schéma IND-CCA bezpečné, je nutně i NM-CPA bezpečné a opět dostáváme spor s předchozím tvrzením.

Závěrem této kapitoly ještě tvrdíme, že generické schéma vytvořené metodou *Encrypt-than-MAC*, jehož schéma autentizace zpráv je silně nefalzifikovatelné, zajišťuje integritu otevřených i šifrových textů a bezpečnost při útocích na nerozlišitelnost s volbou otevřených i širových textů a při útocích na nepozměnitelnost s volbou otevřených i šifrových textů.

Věta 10 *Nechť* $\mathcal{SE} = (\mathcal{K}_e, \mathcal{E}, \mathcal{D})$ *je symetrické šifrovací schéma,* $\mathcal{MA} = (\mathcal{K}_m, \mathcal{T}, \mathcal{V})$ *nechť je schéma autentizace zpráv a ať* $\overline{\mathcal{SE}} = (\overline{\mathcal{K}}, \overline{\mathcal{E}}, \overline{\mathcal{D}})$ *je kombinované schéma získané pomocí metody Encrypt-than-MAC. Pak pro jakéhokoli daného útočníka* I*, útočícího na* $\overline{\mathcal{SE}}$*, můžeme sestrojit útočníka* F *tak, že:*

$$\mathbf{Adv}^{\mathrm{int-ctxt}}_{\overline{\mathcal{SE}}}(I) \leq \mathbf{Adv}^{\mathrm{suf-cma}}_{\mathcal{MA}}(F).$$

Důkaz: Útočníka F sestrojíme stejně jako v případě věty 9. Dále předpokládáme, že $C = C' \parallel \tau'$ je takový dotaz útočníka I na verifikační orákulum, který vede k jeho vítězství ve hře INT-CTXT$_{\overline{\mathcal{SE}}}$. To znamená, že C nebyl útočníkovi vrácen jako odpověď šifrovacího orákula, tudíž C' nebyla útočníkem F nikdy poslána jako dotaz na jeho tagovací orákulum. Odsud již zřejmě vyplývá, že F zvítězil ve hře SUF-CMA$_{\mathcal{MA}}$.

■

Všechny ostatní vlastnosti již můžeme snadno odvodit. Věta 8 nám říká, že schéma vytvořené metodou *Encrypt-than-MAC*, jehož schéma autentizace zpráv je slabě nefalzifikovatelné, zajišťuje IND-CPA bezpečnost. To jistě platí i při použití schématu, jež je silně nefalzifikovatelné. Dle věty 10 zajišťuje takové kombinované schéma INT-CTXT bezpečnost, tedy jistě i INT-PTXT bezpečnost. Dále platí, že je-li schéma zároveň INT-CTXT a IND-CPA bezpečné, je i IND-CCA bezpečné a tato vlastnost dále zajišťuje, že je i NM-CPA bezpečné.

Kapitola 4

Použití hašovacích funkcí s klíčem při autentizaci zpráv

Tato kapitola, ve které si představíme využití kryptografických hašovacích funkcí iterativního typu při autentizaci zpráv, čerpá z článku [2]. Narozdíl od MACů zkonstruovaných na základě blokových šifer jsou hašovací funkce jednoduché, rychlé a dostupné. Tyto funkce však nijak nezávisí na tajném klíči, a proto je třeba je k těmto účelům upravit[1]. Ukážeme si schéma NMAC, ve kterém lze použít jakoukoli hašovací funkci iterativního typu a které je zároveň rychlé, bezpečné a praktické.

4.1 Úvodní poznámky

Zavedení klíče do hašovací funkce. Hašovací funkce nebyly původně navrženy se závislostí na klíči. Proto, chceme-li je využít ve schématech autentizace zpráv, musíme nejprve tuto závislost zavést. Využijeme k tomu inicializační vektor (IV), jehož hodnota bývá v klasických hašovacích funkcích obvykle fixovaná. Tuto hodnotu nahradíme náhodným řetězcem, který bude sloužit jako tajný klíč.

Bezpečnost. Bezpečnost nově vzniklého schématu je zcela závislá na hašovací funkci, jež je použita jako jeho základ. Pokud se prokáže, že schéma je v nějakém ohledu slabé, znamená to, že je slabá příslušná hašovací funkce. Ukážeme ovšem, že bezpečnostní požadavky na tuto funkci nemusí být příliš vysoké a schéma je tedy použitelné v praxi. Při použití náhodného klíče místo fixovaného inicializačního vektoru totiž

[1]Hodnota MACu se vytváří nejen v závislosti na podobě zprávy, ale také v závislosti na klíči. Právě znalost tohoto klíče umožňuje ověření autentizace.

útočník musí komunikovat s oprávněným uživatelem funkce, což neumožňuje paralelizovat tradiční narozeninový útok.

Hašovací funkce jako black-box. Protože je hašovací funkce použita ve schématu formou black-boxu, není třeba uvažovat žádné závislosti ani charakteristické rysy příslušné funkce. Ta může být navíc dle potřeby nahrazena funkcí jinou.

Bezpečný MAC. Bezpečnost MACu vyjádříme pomocí pravděpodobnosti úspěchu útočníka. Tato pravděpodobnost je funkcí počtu platných MACů, které útočník dostane k dispozici (q), a dostupného výpočetního času (t). Pro dané q a t závisí na parametrech schématu, zejména na délce klíče.

Definice 1 *Řekneme, že MAC je (ϵ, t, q, L)-bezpečný, pokud se libovolnému útočníkovi, který nezná hodnotu klíče k a který je omezen celkovým výpočetním časem t a celkovým počtem q dotazů na hodnotu funkce MAC_k, každý z nich maximální délky L, útok na tento MAC podaří s pravděpodobností nejvýše ϵ.*

Do celkového výpočetního času t zahrnujeme i čas potřebný pro výpočet hodnoty funkce MAC_k pro každý z dotazů. Tento přístup velmi dobře vystihuje fakt, že čas potřebný pro výpočet odpovědí na tyto dotazy, obzvláště pokud je jejich počet vysoký, může mít značný podíl na časové složitosti útoku. Dále do tohoto času zahrnujeme i velikost kódu útočníkova algoritmu.

4.2 Kryptografické hašovací funkce

Základní vlastnosti. Vstupem kryptografické hašovacích funkce je řetězec libovolné délky, výstupem pak řetězec pevně dané délky. Tyto funkce jsou navrženy především tak, aby byly odolné vůči kolizím. To znamená, že máme-li dánu funkci F, pak by pro útočníka mělo být neproveditelné nalézt dvojici různých řetězců x, x' takových, že $F(x) = F(x')$. Dále by tato funkce měla vypadat náhodně (nezávislost na vstupu a výstupu, nepředvídatelnost výstupu atd.).

Iterativní konstrukce. Tato metoda konstruování hašovacích funkcí je založena na základní jednotce zvané *kompresní funkce*. Ta zpracovává krátké, stejně dlouhé části vstupu a je iterována. Tímto způsobem je pak získán haš libovolně dlouhého řetězce.

Kompresní funkci budeme značit symbolem f. Funkce má dva vstupy, *řetězící proměnnou (chaining variable)* délky l a blok dat délky b. Hašování pak probíhá následovně: Nejprve zafixujeme hodnotu IV délky l bitů. Vstupní data pak převedeme do tvaru $x = x_1, x_2, \ldots, x_n$, kde počet bloků n je libovolné číslo a každý z bloků x_i má délku b bitů (pokud velikost vstupu není dělitelná číslem b, použijeme padding). Dále položíme hodnotu bloku $x_{n+1} = |x|$ a výslednou hodnotu iterativní funkce označíme F. Platí, že $F(x) = h_{n+1}$, kde $h_0 = IV$ a $h_i = f(h_{i-1}, x_i)$ pro $i = 1, 2, \ldots, n+1$.

4.3 Hašovací funkce s klíčem

Jak již bylo uvedeno, podstatným prvkem funkcí na autentizaci zpráv je tajný klíč. Většina kryptografických hašovacích funkcí ale žádný klíč nevyužívá, proto nejprve musíme definovat, jakým způsobem hašovací funkci společně s klíčem použít.

Nejběžnějším postupem bývá zakomponování klíče do dat, která chceme hašovat (např. klíč a data zřetězíme a poté hašujeme). V našem případě ale postupujeme tak, jak již bylo naznačeno v úvodu kapitoly. Fixovanou hodnotu inicializačního vektoru původní funkce nahradíme tajným klíčem, jehož hodnotu budou znát pouze komunikující strany. Uvidíme, že tento přístup má značné výhody. Při použití této metody můžeme hašovací funkce s klíčem uvažovat jako *rodinu funkcí.*

Kompresní funkci s klíčem označíme f_k, kde $f_k(x) = f(k, x)$ pro $|k| = l$ a $|x| = b$. S libovolnou iterativní hašovací konstrukcí asociujeme rodinu iterativních hašovacích funkcí s klíčem $\{F_k\}_k$ tak, že pro $x = x_1 \ldots x_n$ definujeme $F_k(x) = k_{n+1}$, kde $k_i = f_{k_{i-1}}(x_i)$ pro $i = 1, \ldots, n+1$, $k_0 = k$ a $x_{n+1} = |x|$. Prostor klíčů je pro kompresní funkce s klíčem a pro iterativní hašovací funkce s klíčem totožný a jedná se o všechny řetězce délky l.

Definice 2 *Říkáme, že rodina iterativních hašovacích funkcí s klíčem $\{F_k\}$ je (ϵ, t, q, L)-slabě odolná vůči kolizi, pokud libovolný útočník, který nezná hodnotu klíče k, jehož celkový výpočetní čas je omezen hodnotou t, a který vidí hodnoty funkce $\{F_k\}$ pro q jím vybraných zpráv $m_1, m_2, \ldots, m_q$, kde délka každé z nich je maximálně L, není schopen nalézt zprávy m, m' takové, že $F_k(m) = F_k(m')$, s pravděpodobností lepší než ϵ.*

Je dobré si uvědomit, že v případě tradičních hašovacích funkcí bez klíče stačí najít kolizi pro známou a neměnnou hodnotu inicializačního vektoru. To znamená, že útočník může pracovat nezávisle na jakémkoli uživateli a klíči a může útok paralelizovat. Naproti tomu u hašovacích

funkcí s tajným klíčem potřebuje útočník interagovat s oprávněným uživatelem, který zná hodnotu klíče. Paralelní útok je tedy vyloučen.

4.4 NMAC

Nyní zavedeme schéma autentizace zpráv NMAC (Nested MAC) a poté dokážeme jeho bezpečnost.

Definice 3 *Mějme $k = (k_1, k_2)$, kde k_1, k_2 jsou klíče pro výpočet funkce F (tj. náhodné řetězce délky l). Definujeme autentizační funkci NMAC(x), jejíž vstup x je libovolně dlouhý, jakožto:*

$$\mathrm{NMAC}_k(x) = F_{k_1}(F_{k_2}(x)).$$

Vidíme, že tato konstrukce je velmi jednoduchá a efektivní. Čas, který je třeba k výpočtu vnitřní funkce je stejný, jako kdybychom použili původní hašovací funkci bez klíče, takže navíc se provede pouze jedna iterace kompresní funkce. Nyní ukážeme, že síla této konstrukce je závislá na kryptografické síle původní hašovací funkce.

Věta 11 *Pokud je kompresní funkce s klíčem f (ϵ_f, q, t, b)-bezpečný MAC použitý na zprávy délky b bitů a pokud iterativní hašovací funkce s klíčem F je (ϵ_F, t, q, L)-slabě odolná vůči kolizím, potom pro funkci NMAC platí, že je $(\epsilon_f + \epsilon_F, q, t, b)$-bezpečný MAC.*

Důkaz: Mějme útočníka A_N, který útočí na funkci NMAC a zafixujme parametry q, t a L, které reprezentují po řadě počet dotazů, výpočetní čas a délku zpráv, jež jsou útočníkovi při útoku k dispozici. Označme pravděpodobnost úspěchu tohoto útočníka jako ϵ_N. Dále nechť ϵ_F je nejlepší možná pravděpodobnost úspěchu libovolného útočníka, který má k dispozici stejné zdroje jako A_N, ve snaze najít kolizi pro funkci F_{k_2} bez znalosti klíče k_2. S využitím útočníka A_N sestrojíme útočníka A_f, který útočí na kompresní funkci f_{k_1} (pro vstupy délky b, s použitím q dotazů, v čase t) s pravděpodobností $\epsilon_f \geq \epsilon_F + \epsilon_N$. Odsud již dostaneme, že libovolný útočník, který se snaží prolomit funkci NMAC s použitím již zmíněných zdrojů, má pravděpodobnost úspěchu ϵ_N nejvýše $\epsilon_f + \epsilon_F$.

Nejprve definujeme symbol $\overline{s}$. Je-li s řetězec délky l, pak $\overline{s}$ získáme tak, že s doplníme pomocí paddingu[2] na blok délky b. Nyní již můžeme

[2]Přesný postup závisí na hašovací funkci, která byla pro schéma NMAC použita.

popsat běh útočníka A_f, který má k dispozici orákula F_{k_2} a f_{k_1}. A_f nejprve zvolí náhodně klíč k_2 a poté jako svůj podprogram spustí útočníka A_N. Tento útočník zasílá dotazy x_i, kde $i = 1, \ldots, q$. Útočník A_f pro každý z dotazů nejprve zjistí hodnotu $F_{k_2}(x_i) = z_i$, tu, je-li třeba[3], doplní na blok délky b (získá tak $\overline{z_i}$) a pošle ji jako dotaz na funkci f_{k_1}. Odpověď na tento dotaz potom vrátí útočníkovi A_N. Poté, co takto A_N dostane odpovědi na všech svých q dotazů, pošle jako svůj výstup dvojici (x, y).[4] Útočník A_f zjistí hodnotu $F_{k_2}(x)$ a jako svůj výstup odpoví dvojici $(\overline{F_{k_2}(x)}, y)$.

Dále nás zajímá, jaká je pravděpodobnost ϵ_f, že útočník A_f při útoku uspěje. Je zřejmé, že neúspěch nastane právě ve dvou případech: 1) neuspěje útočník A_N; 2) útočník A_N sice ve svém útoku uspěje, ale platí, že $\overline{F_{k_2}(x)} = \overline{F_{k_2}(x_i)}$ pro nějaké $i \in \{1 \ldots q\}$. Z popisu algoritmu vidíme, že útočník A_N získává takové odpovědi, jako by opravdu útočil na funkci NMAC, takže pravděpodobnost jeho neúspěchu je stejná, jako kdyby útočil na NMAC_k, tedy nanejvýš $1 - \epsilon_N$. Pro druhý případ si stačí uvědomit, že pokud by A_f na začátku svého běhu místo klíče k_2 zvolil náhodně klíč k_1, mohl by pomocí dotazů x_i útočit na funkci F_{k_2}. Případ dvě by pak znamenal, že byla nalezena kolize pro funkci F_{k_2}, protože původní rovnost jistě můžeme převést na tvar $F_{k_2}(x) = F_{k_2}(x_i)$. Pravděpodobnost nalezení takové kolize je ale shora ohraničena hodnotou ϵ_F.

Pokud obě hodnoty sečteme, získáme tak, že $\epsilon_f \leq 1 - (1 - \epsilon_N + \epsilon_F)$, odkud již jasně plyne, že $\epsilon_N \leq \epsilon_f + \epsilon_F$ a důkaz je u konce.

■

[3]Připouštíme možnost, že ve schématu mohou být použity různé hašovací funkce, tedy velikost jejich výstupu nemusí být totožná.

[4]Připomeňme, že A_N útočí na funkci $\mathrm{NMAC}_k, k = (k_1, k_2)$, tedy se snaží najít takovou dvojici, kde x je zpráva různá od všech jím dotazovaných x_i a $y = \mathrm{NMAC}_k(x)$.

Kapitola 5

Schémata šifrování s ochranou integrity s připojenými daty

V praxi se často setkáme s problémem, kdy potřebujeme zajistit utajení pouze pro určitou část dat a zbylá data posíláme v nezměněné podobě (například hlavičku zprávy). Pro obojí ale potřebujeme zajistit autentizaci. V této kapitole, založené na článku [3], ukážeme řešení této situace, a to schémata šifrování s ochranou integrity s připojenými daty (AEAD)[1]. Základem této metody je použití již existujícího schématu šifrování s ochranou integrity, které se pouze vhodně upraví. Představíme dva různé postupy, *nonce stealing* a *ciphertext translation*.

5.1 Základní definice

Útočník respektující nonce. Útočník respektující nonce je takový útočník, který při dotazu na své orákulum nikdy nezopakuje nonci N, nehledě na předchozí odpovědi orákula. V celé kapitole budeme nadále předpokládat, že každý útočník, který útočí na AE- nebo AEAD-schéma, respektuje nonce.

AE-schéma užívající nonci. AE-schéma je trojice $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$. Přidružené k Π jsou množiny Nonce $= \{0,1\}^n$ a Message $\subseteq \{0,1\}^*$. Test náležení do množiny Message probíhá v lineárním čase a pro tuto množinu platí $M \in$ Message $\Rightarrow M' \in$ Message pro libovolné M', které má stejnou délku jako M. Prostor klíčů $\mathcal{K}$ je konečná a neprázdná množina řetězců. Algoritmus $\mathcal{E}$ je deterministický, na vstupu má řetězce $K \in \mathcal{K}$,

[1] Zkratka je odvozena z anglického termínu authenticated encryption with associated data.

$N \in$ Nonce a $M \in$ Message a vrací řetězec $\mathcal{C} = \mathcal{E}_K^N(M) = \mathcal{E}_K(N, M)$. Algoritmus $\mathcal{D}$ je deterministický, na vstupu má řetězce $K \in \mathcal{K}$, $N \in$ Nonce a $\mathcal{C} \in \{0,1\}^*$ a vrací $\mathcal{D}_K^N(\mathcal{C})$, což je buď nějaký řetězec z množiny Message nebo symbol $\perp$ pro neplatný text. Požadujeme, aby platilo $\mathcal{D}_K^N(\mathcal{E}_K^N(M)) = M$ pro všechny $K \in \mathcal{K}$, $N \in$ Nonce a $M \in$ Message. Dále předpokládáme, že $|\mathcal{E}_K^N(M)| = \ell(|M|)$, kde ℓ je nějaká "funkce délky" vypočitatelná v lineárním čase.

Nechť $\$(\cdot,\cdot)$ je orákulum, které pro dotaz (N, M) vrací jako odpověď náhodný řetězec délky $\ell(|M|)$, kde ℓ je funkce délky pro schéma Π, a nechť A je útočník. Pak definujeme:

$$\mathbf{Adv}_{\Pi}^{\mathrm{priv}}(A) = \Pr[K \xleftarrow{\$} \mathcal{K} : A^{\mathcal{E}_K(\cdot,\cdot)} = 1] - \Pr[A^{\$(\cdot,\cdot)} = 1].$$

Tento pojem nazýváme *nerozlišitelnost od náhodných bitů při útoku s volbou otevřených textů* (IND$-CPA).

Nechť $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ je schéma šifrování s ochranou integrity. Zvolíme náhodně K z $\mathcal{K}$ a spustíme útočníka A, který bude mít k dispozici šifrovací orákulum $\mathcal{E}_K(\cdot,\cdot)$. Řekneme, že A *padělá*, pokud se mu podaří nalézt dvojici $(N, \mathcal{C})$, kde $\mathcal{D}_K^N(\mathcal{C}) \neq \perp$ a zároveň se A nezeptal na dotaz $\mathcal{E}(N, M)$, který vedl k odpovědi $\mathcal{C}$. Pravděpodobnost, že A padělá, je pak $\mathbf{Adv}_{\Pi}^{\mathrm{auth}}(A)$. Tuto pravděpodobnost uvažujeme přes všechny náhodné volby K.

Hašovací funkce AXU (almost-xor-universal). Mějme rodinu funkcí $F : \mathcal{K} \times \mathcal{X} \to \{0,1\}^\tau$ a útočníka A. Potom

$$\mathbf{Adv}_F^{\mathrm{axu}}(A) = \max\{\delta, \epsilon\},$$

kde pro hodnoty δ a ϵ platí:

$$\begin{aligned} \delta &= \Pr[K \xleftarrow{\$} \mathcal{K}; (X_1, X_2, \Delta) \leftarrow A : X_1 \neq X_2 \ \&\ F_K(X_1) \oplus F_K(X_2) = \Delta], \\ \epsilon &= \Pr[K \xleftarrow{\$} \mathcal{K}; (X, C) \leftarrow A : F_K(X) = C]. \end{aligned}$$

Pseudonáhodné funkce. Nechť $F : \mathcal{K} \times \mathcal{X} \to \{0,1\}^\tau$ je rodina funkcí a $\mathrm{Rand}(\mathcal{X}, \tau)$ je množina všech funkcí z $\mathcal{X}$ do $\{0,1\}^\tau$. Pro libovolného útočníka A definujeme:

$$\mathbf{Adv}_F^{\mathrm{prf}}(A) = \Pr[K \xleftarrow{\$} \mathcal{K} : A^{F_K(\cdot)} = 1] - \Pr[\rho \xleftarrow{\$} \mathrm{Rand}(\mathcal{X}, \tau) : A^{\rho}(\cdot) = 1].$$

5.2 Zavedení AEAD-schématu

AEAD-schéma je trojice $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$. Přidružené k Π jsou pak množiny Nonce $= \{0,1\}^n$, Message $\subseteq \{0,1\}^*$ a Header $\subseteq \{0,1\}^*$, pro kterou test náležení do množiny probíhá v lineárním čase. Prostor klíčů $\mathcal{K}$ je konečná a neprázdná množina řetězců. Šifrovací algoritmus $\mathcal{E}$ je deterministický, na vstupu má řetězce $K \in \mathcal{K}$, $N \in$ Nonce, $H \in$ Header a $M \in$ Message. Jeho výstupem je pak řetězec $\mathcal{C} = \mathcal{E}_K^{N,H}(M) = \mathcal{E}_K(N, H, M)$. Dešifrovací algoritmus $\mathcal{D}$ je také deterministický, na vstupu má řetězce $K \in \mathcal{K}$, $N \in$ Nonce, $H \in$ Header a $\mathcal{C} \in \{0,1\}^*$ a vrací $\mathcal{D}_K^{N,H}(\mathcal{C})$, což je buď nějaký řetězec z množiny Message nebo symbol $\perp$ pro neplatný text. Samozřejmě požadujeme, aby pro všechny řetězce $K \in \mathcal{K}$, $N \in$ Nonce, $H \in$ Header a $M \in$ Message platilo $\mathcal{D}_K^{N,H}(\mathcal{E}_K^{N,H}(M)) = M$, a aby pro nějakou funkci délky ℓ vypočitatelnou v lineárním čase platilo $|\mathcal{E}_K^N(M)| = \ell(|M|)$.

Nechť $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ je AEAD-schéma s funkcí délky ℓ, A je útočník a $\$(\cdot,\cdot,\cdot)$ nechť je orákulum, které na dotaz (N, H, M) vrací náhodný řetězec délky $\ell(|M|)$ bitů. Opět definujeme IND$-CPA vlastnost, tentokráte pro AEAD-schéma[2], následovně:

$$\mathbf{Adv}_{\Pi}^{\mathrm{PRIV}}(A) = \Pr[K \xleftarrow{\$} \mathcal{K} : A^{\mathcal{E}_K(\cdot,\cdot,\cdot)} = 1] - \Pr[A^{\$(\cdot,\cdot,\cdot)} = 1].$$

Dále nechť B je útočník, který má přístup k orákulu $\mathcal{E}_K(\cdot,\cdot,\cdot)$ pro nějaký klíč K. Opět řekneme, že B *padělá*, pokud nalezne trojici $(N, H, \mathcal{C})$ takovou, že $\mathcal{D}_K^{N,H}(\mathcal{C}) \neq \perp$ a zároveň pokud B nepoložil orákulu dotaz (N, H, M), který vedl k odpovědi $\mathcal{C}$. Pravděpodobnost, že B padělá, definujeme jako $\mathbf{Adv}_{\Pi}^{\mathrm{AUTH}}(B)$[3]. Tuto pravděpodobnost uvažujeme přes všechny volby klíče K.

5.3 Nonce stealing

První metodou, jak z daného AE-schématu vytvořit AEAD-schéma, je *nonce stealing*. Využijeme toho, že aplikace používající dané AE-schéma vystačí s kratší noncí, než schéma nabízí. Zbývající místo tedy můžeme vyplnit připojenými daty.

Formálněji řečeno, máme-li dáno AE-schéma $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$, které má prostor noncí Nonce $= \{0,1\}^n$, a máme-li dán parametr $\bar{n} = 1, \ldots, n-1$, pak definujeme AEAD-schéma $\bar{\Pi} = \Pi|\bar{n} = (\bar{\mathcal{K}}, \bar{\mathcal{E}}, \bar{\mathcal{D}})$ s prostorem noncí Nonce $= \{0,1\}^{\bar{n}}$ a s prostorem připojených dat Header $= \{0,1\}^{n-\bar{n}}$, kde $\bar{\mathcal{K}} = \mathcal{K}$, $\bar{\mathcal{E}}_K^{N,H}(M) = \mathcal{E}_K^{N\|H}(M)$ a $\bar{\mathcal{D}}_K^{N,H}(\mathcal{C}) = \mathcal{D}_K^{N,H}(\mathcal{C})$.

[2]Rozlišujeme tedy $\mathbf{Adv}^{\mathrm{PRIV}}$ pro AEAD-schéma a $\mathbf{Adv}^{\mathrm{priv}}$ pro AE-schéma.

[3]Opět si všimněme použití velkých písmen v definici pro AEAD-schéma.

V následující větě dokážeme bezpečnost AEAD-schématu vytvořeného metodou nonce stealing. Parametry t, q, σ, ρ představují po řadě výpočetní čas, počet dotazů, maximální délku všech dotazů celkem a maximální délku výstupu útočníka, jež jsou při útoku k dispozici. Uvedené hodnoty advantage jsou potom nejlepší možné, uvažujeme-li všechny útočníky, kteří jsou omezeni danými zdroji[4].

Věta 12 *Nechť* Π *je AE-schéma s prostorem noncí* $\text{Nonce} = \{0,1\}^n$ *a nechť* $\bar{n} \in [1 \ldots n]$. *Potom:*

$$\begin{aligned} \mathbf{Adv}^{\text{PRIV}}_{\Pi|\bar{n}}(t,q,\sigma) &\leq \mathbf{Adv}^{\text{priv}}_{\Pi}(t_1,q,\sigma), \\ \mathbf{Adv}^{\text{AUTH}}_{\Pi|\bar{n}}(t,q,\sigma,\rho) &\leq \mathbf{Adv}^{\text{auth}}_{\Pi}(t_1,q,\sigma,\rho), \end{aligned}$$

kde $t = t_1 + O(\sigma + q)$ *a* $t = t_2 + O(\sigma + \rho + q)$.

Důkaz: Nejprve dokážeme první nerovnost, jež se týká utajení. Mějme útočníka A, který útočí na utajení AEAD-schématu $\bar{\Pi} = \Pi|\bar{n}$. Nyní můžeme sestrojit útočníka B, který bude útočit na utajení AE-schématu Π s využitím útočníka A. Útočník B spustí útočníka A. Ten posílá dotazy tvaru (N_i, H_i, M_i), které B převede do tvaru $(N_i \parallel H_i, M_i)$ a pošle je svému orákulu. Jeho odpověď $\mathcal{C}_i$ potom pošle zpět útočníkovi A. Na konci běhu A odpoví svůj odhad na hodnotu bitu b, B potom odpoví tu samou hodnotu.

Jak jsme již zmínili dříve, předpokládáme, že každý útočník útočící na AEAD-schéma respektuje nonce, tedy pro všechny dotazy (N_i, H_i, M_i) útočníka A platí, že hodnoty N_i jsou různé. To ale znamená, že i všechny hodnoty $N_i \parallel H_i$, které zasílá útočník B, jsou různé. Vidíme, že A získává odpovědi, které očekává, tedy jeho rozhodování není ovlivněno. Pokaždé, když A uhodne hodnotu b, znamená to, že ji uhodne i B (jejich advantage se rovnají) a nerovnost je tak dokázána.

Pro důkaz autenticity znovu použijeme oba útočníky. Nyní A útočí na autenticitu schématu $\bar{\Pi}$. Opět sestrojíme útočníka B, který bude útočit na autenticitu Π. Znovu B pouze spustí A, jeho dotazy tvaru (N_i, H_i, M_i) převede na tvar $(N_i \parallel H_i, M_i)$, pošle svému orákulu a odpověď vrátí útočníkovi A. A nyní na konci běhu pošle svůj pokus o padělek $(N, H, \mathcal{C})$. B jej převede na tvar $(N \parallel H, \mathcal{C})$, který pošle jako svůj pokus o padělek.

Opět vidíme, že A nepozná, že komunikuje s B namísto svého orákula, tedy padělek vyrábí nezávisle. Pokud je jeho padělek úspěšný, je úspěšný i padělek útočníka B a dokázali jsme tak i druhou nerovnost.

■

[4]Jedná se vlastně o pojem InSecurity.

5.4 Ciphertext translation

Druhá metoda, zvaná *ciphertext translation*, je poněkud náročnější. Umožňuje nám ale připojit data libovolné délky. K převedení AE-schématu na AEAD-schéma nyní využijeme rodinu funkcí $F : \mathcal{K}' \times \text{Header} \to \{0,1\}^\tau$.

Mějme AEAD-schéma $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$, ve kterém je minimální délka šifrového textu rovna nějaké konstantě τ. Dále nechť $\text{Header} \subseteq \{0,1\}^*$ je množina řetězců, pro niž test náležení probíhá v lineárním čase, a nechť $F : \mathcal{K}' \times \text{Header} \to \{0,1\}^\tau$ je rodina funkcí. Jednotlivé transformace AEAD-schématu $\ddot{\Pi} = \Pi \cdot\cdot F = (\ddot{\mathcal{K}}, \ddot{\mathcal{E}}, \ddot{\mathcal{D}})$ definujeme následovně:

$$\ddot{\mathcal{K}} = \mathcal{K} \times \mathcal{K}',$$

$$\ddot{\mathcal{E}}^{N,H}_{KK'}(M) = \mathcal{E}^N_K(M) \mathbin{\widehat{\oplus}} F_{K'}(H),^5$$

$$\ddot{\mathcal{D}}^{N,H}_{KK'}(\mathcal{C}) = \mathcal{D}^N_K(\mathcal{C} \mathbin{\widehat{\oplus}} F_{K'}(H)).$$

Nyní ukážeme, že i takto vytvořené AEAD-schéma je bezpečné, pokud je bezpečné původní AE-schéma i funkce F. Parametry použité pro hodnoty advantage jsou stejné jako v předchozím případě a opět uvažujeme nejlepší možné hodnoty přes všechny útočníky omezené danými zdroji. Pro autenticitu uvedeme dva vztahy. V prvním použijeme jako funkci F pseudonáhodnou funkci, ve druhém případě použijeme hašovací funkci AXU.

Věta 13 *Mějme AE-schéma $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$, pro nějž platí, že každý šifrový text je dlouhý alespoň τ bitů. Nechť $F : \mathcal{K}' \times \text{Header} \to \{0,1\}^\tau$ je rodina funkcí. Potom:*

$$\begin{aligned}
\mathbf{Adv}^{\text{PRIV}}_{\Pi\text{-}F}(t,q,\sigma) &\leq \mathbf{Adv}^{\text{priv}}_{\Pi}(t_1,q,\sigma), && (5.1)\\
\mathbf{Adv}^{\text{AUTH}}_{\Pi\text{-}F}(t,q,\sigma,\widehat{\sigma},\rho) &\leq \mathbf{Adv}^{\text{auth}}_{\Pi}(t_2,q,\sigma,\rho) + \mathbf{Adv}^{\text{priv}}_{\Pi}(t_3,q,\sigma) + \\
&\quad + \mathbf{Adv}^{\text{prf}}_{\Pi}(t_4,2,\widehat{\sigma}+\rho) + 2^{-\tau}, && (5.2)\\
\mathbf{Adv}^{\text{AUTH}}_{\Pi\text{-}F}(t,q,\sigma,\widehat{\sigma},\rho) &\leq \mathbf{Adv}^{\text{auth}}_{\Pi}(t_5,q,\sigma,\rho) + \mathbf{Adv}^{\text{priv}}_{\Pi}(t_6,q,\sigma) + \\
&\quad + \mathbf{Adv}^{\text{axu}}_{\Pi}(\widehat{\sigma} + \rho), && (5.3)
\end{aligned}$$

kde $t = t_1 + \text{Time}_F(q,\sigma) + O(\sigma+q)$, $t = t_2 + t_3 + t_4 + 2 \cdot \text{Time}_F(q+1, \sigma+\rho) + O(\sigma+\rho+q)$ *a* $t = t_5 + t_6 + 2 \cdot \text{Time}_F(q+1,\sigma+\rho) + O(\sigma+\rho+q)$.

[5] Symbol $\oplus$ znamená, že kratší z řetězců doplníme zepředu nulami na délku toho druhého a poté přixorujeme.

Důkaz: Předpokládejme, že máme útočníka A, který útočí na utajení AEAD-schématu $\ddot{\Pi} = \Pi \cdot\cdot F = (\ddot{\mathcal{K}}, \ddot{\mathcal{E}}, \ddot{\mathcal{D}})$ a který běží v čase maximálně t, položí nanejvýš q dotazů celkové délky maximálně σ. Potom můžeme sestrojit útočníka B, který bude útočit na utajení AE-schématu Π. B nejprve vybere náhodný klíč K' z $\mathcal{K}'$ a poté spustí útočníka A. Pokaždé, když A pošle dotaz tvaru (N_i, H_i, M_i) kde $i = 1, \ldots, q$, B pošle svému orákulu dotaz tvaru (N_i, M_i). Zpátky získá odpověď $\mathcal{C}_i$, vypočte $\Delta_i = F_{K'}(H_i)$ a jako odpověď pro útočníka A pak pošle $\ddot{\mathcal{C}}_i = \mathcal{C}_i \mathbin{\widehat{\oplus}} \Delta_i$. Na konci svého běhu A odpoví svůj odhad na hodnotu bitu b. Stejnou hodnotu poté odpoví i útočník B.

Vidíme, že A získává přesně takové odpovědi, jaké očekává, tedy jeho rozhodování není nikterak ovlivněno. Uhodne-li správně hodnotu bitu b, uhodne ji správně i útočník B (opět mají shodné hodnoty advantage) a dostáváme tedy nerovnost 5.1.

Dále mějme znovu útočníka A, který tentokrát bude útočit na autenticitu schématu $\Pi \cdot\cdot F$. Opět předpokládáme, že běží v čase maximálně t, položí nejvýše q dotazů, každý z nich dlouhý nejvýš $\widehat{\sigma}$, celkové délky maximálně σ a jehož výstup je omezen hodnotou ρ. Sestrojíme útočníky A_{auth}, A_{priv} a A_{prf} útočící po řadě na autenticitu Π, utajení Π a na funkci F, která je v tomto případě PRF.

Začneme konstrukcí útočníka A_{auth}. Ten na začátku svého běhu zvolí náhodný klíč K' z množiny $\mathcal{K}'$ a poté spustí útočníka A. Když A položí dotaz tvaru (N_i, H_i, M_i), $i = 1, \ldots, q$, B svému orákulu pošle dotaz (N_i, M_i) a dostane odpověď $\mathcal{C}_i$. Dále vypočte $\Delta_i = F_{K'}(H_i)$ a jako odpověď pro útočníka A pošle $\ddot{\mathcal{C}}_i = \mathcal{C}_i \mathbin{\widehat{\oplus}} \Delta_i$. Po zaslání všech dotazů A pošle svůj pokus o padělek $(N, H, \mathcal{C})$. B vypočítá hodnoty $\Delta = F_{K'}(H)$ a $\mathcal{C}^* = \mathcal{C} \mathbin{\widehat{\oplus}} \Delta$ a jako svůj pokus o padělek pošle dvojici $(N, \mathcal{C}^*)$.

Opět vidíme, že A není ve svém rozhodování nikterak ovlivněn, tedy jeho advantage je stejná, jako kdyby útočil na schéma $\Pi \cdot\cdot F$. Ovšem narozdíl od předchozího případu se může stát, že ačkoli útočník A nalezne padělek, tedy trojici $(N, H, \mathcal{C})$ takovou, že dešifrováním $\mathcal{C}$ vznikne platný otevřený text M a (N, H, M) nebyl předem poslán jako dotaz, útočník A_{auth} po přixorování hodnoty Δ k posledním τ bitům $\mathcal{C}$ může dostat takové $\mathcal{C}^*$, které již s hodnotou N poslal jako dotaz. Jeho pokus o padělek tedy nebude platný a jeho advantage je nižší než u útočníka A.

Definujeme tedy nyní událost *koliduje*. Útočník A poslal pokus o padělek $(N, H, \mathcal{C})$, kde $\mathcal{C} = C \parallel T$, $|T| = \tau$ potom, co dostal odpovědi $C_i \parallel T_i$, kde $N = N_i$ pro nějaké $i = 1, \ldots, q$. Aby následný pokus o padělek $(N, \mathcal{C}^*)$ útočníka A_{auth} byl neplatný, musí platit $C = C_i$ a $T \oplus F_{K'}(H) = T_i \oplus F_{K'}(H_i)$, což můžeme přepsat jako $T \oplus T_i = F_{K'}(H) \oplus F_{K'}(H_i)$.

Událost *koliduje* tedy nastane, platí-li pro pokus o padělek $(N, H, \mathcal{C})$ útočníka A a pro nějaké $i = 1, \ldots, q$, že $N = N_i$, $C = C_i$ a $T \oplus T_i = F_{K'}(H) \oplus F_{K'}(H_i)$. Pak platí, že:

$$\mathbf{Adv}^{\mathrm{AUTH}}_{\Pi\text{-}F}(A) = \Pr[A^{\Pi\text{-}F}\text{padělá}] \leq \Pr[A^{\Pi}_{\mathrm{auth}}\text{padělá}] \ + \ \Pr[A^{\Pi\text{-}F}\text{koliduje}].$$

Pravděpodobnost, že útočník A *koliduje*, pak můžeme přepsat následovně:

$$\begin{aligned}\Pr[A^{\Pi\text{-}F}\text{koliduje}] \ = \ & (\Pr[A^{\Pi\text{-}F}\text{koliduje}] \ - \Pr[A^{\$\text{-}F}\text{koliduje}]) + \\ & (\Pr[A^{\$\text{-}F}\text{koliduje}] \ - \ \Pr[A^{\$\text{-}R}\text{koliduje}]) \ + \\ & \Pr[A^{\$\text{-}R}\text{koliduje}]\end{aligned}$$

Symbol \$ značí orákulum, které při dotazu tvaru (N, H, M) vrací náhodný řetězec bitů délky $\ell(|M|)$, a R je funkce z množiny $\mathrm{Rand}(\{0,1\}^*, \tau)$. Ukážeme, že platí:

$$\Pr[A^{\Pi\text{-}F}\text{koliduje}] - \Pr[A^{\$\text{-}F}\text{koliduje}] \ \leq \ \mathbf{Adv}^{\mathrm{priv}}_{\Pi}(t_3, q, \sigma, \widehat{\sigma}), \quad (5.4)$$

$$\Pr[A^{\$\text{-}F}\text{koliduje}] - \Pr[A^{\$\text{-}R}\text{koliduje}] \ \leq \ \mathbf{Adv}^{\mathrm{prf}}_{\Pi}(t_4, 2, \widehat{\sigma} + \rho), \quad (5.5)$$

$$\Pr[A^{\$\text{-}R}\text{koliduje}] \ \leq \ 2^{-\tau}. \quad (5.6)$$

Sestrojíme nyní útočníka A_{priv} a ukážeme platnost nerovnosti 5.4. A_{priv} začne svůj běh náhodnou volbou klíče K' z $\mathcal{K}'$ a poté spustí útočníka A. Ten stejně jako v předchozích případech posílá dotazy tvaru (N_i, H_i, M_i), $i = 1, \ldots, q$. A_{priv} pak pro každé i posílá svému orákulu dotaz (N_i, M_i) a získává odpověď $\mathcal{C}_i$. Dále vždy vypočte hodnotu $\Delta_i = F_{K'}(H_i)$ a útočníkovi A potom posílá odpověď $\ddot{\mathcal{C}}_i = \mathcal{C}_i \mathbin{\widehat{\oplus}} \Delta_i$. Po skončení svého běhu A odpoví svůj pokus o padělek (N, H, C). Útočník A_{priv} nyní vypočte, zda pro tento pokus nastala událost *koliduje*, tedy zda pro nějaké i nastane $N = N_i$, $C = C_i$ a $T \oplus T_i = F_{K'}(H) \oplus F_{K'}(H_i)$. Pokud ano, posílá odhad na bit $b = 1$, v opačném případě pak pošle $b = 0$. Vidíme, že pro takto definovaného útočníka platí:

$$\mathbf{Adv}^{\mathrm{priv}}_{\Pi}(A_{\mathrm{priv}}) = \Pr[A^{\Pi\text{-}F}\text{koliduje}] - \Pr[A^{\$\text{-}F}\text{koliduje}].$$

Také vidíme, že běží v čase $t_3 = t + \mathrm{Time}_F(q + 1, \sigma + \rho) + O(\sigma + \rho + q)$. Nerovnost 5.4 tedy platí.

Abychom ukázali nerovnost 5.5, sestrojíme útočníka A_{prf}, který opět spustí útočníka A. Na jeho dotazy (N_i, H_i, M_i) odpovídá hodnotami $\ddot{\mathcal{C}}_i$, což jsou v tomto případě náhodné řetězce bitů délky $\ell(|M_i|)$. Poté, co A ukončí běh a pošle pokus o padělek $(N, H, \mathcal{C})$, A_{prf} pošle svému orákulu dva dotazy, H a H_i, dostane zpět odpovědi Δ a Δ_i a opět zjistí, zda

nastala událost *koliduje*. Pokud ano, vrací hodnotu $b = 1$, pokud ne, vrací hodnotu $b = 0$. Pro takto definovaného útočníka dostáváme:

$$\mathbf{Adv}_F^{\mathrm{prf}}(A_{\mathrm{prf}}) = \Pr[A^{\$\text{-}F}\text{koliduje}] - \Pr[A^{\$\text{-}R}\text{koliduje}].$$

Navíc útočník A_{prf} běží v čase $t_4 = t + O(\sigma + \rho + q)$.

Pro ověření nerovnosti 5.6 si rozmyslíme, kdy A vyprodukuje kolizi, komunikuje-li s orákulem $\$ \cdot\cdot R$. A musí najít taková $T_i,\ H_{i,},\ T,\ H$, aby $T \oplus T_i = R_{K'}(H) \oplus R_{K'}(H_i)$. To znamená, že A musí bez pokládání jakýchkoli dotazů předpovědět pro orákulum $R_{K'}(\cdot)$ hodnotu $R_{K'}(H) \oplus R_{K'}(H_i)$ pro vybraná $H,\ H_i$, nebo hodnotu $R_{K'}(H)$ pro vybrané H. To se mu ale podaří s pravděpodobností $2^{-\tau}$ a dokázali jsme tak nejen 5.6, ale i 5.2.

U důkazu nerovnosti 5.3 budeme postupovat stejně jako v případě 5.2 až do rozepsání hodnoty $\Pr[A^{\Pi\text{-}F}$ koliduje]. Ta nám nyní stačí v této formě:

$$\begin{aligned}\Pr[A^{\Pi\text{-}F}\text{koliduje}] \;=\; & (\Pr[A^{\Pi\text{-}F}\text{koliduje}] - \Pr[A^{\$\text{-}F}\text{koliduje}]) + \\ & + \Pr[A^{\$\text{-}F}\text{koliduje}].\end{aligned}$$

Pro rozdíl v závorce platí stejné omezení jako v předešlém případě, stačí tedy už jen ukázat, že $\Pr[A^{\$\text{-}F}$ koliduje$] \leq \mathbf{Adv}_F^{\mathrm{axu}}(\widehat{\sigma}+\rho)$. Opět si rozebereme situaci, kdy A vyprodukuje kolizi, komunikuje-li s orákulem $\$ \cdot\cdot F$. Stejně jako v předchozím případě musí najít $T_i,\ H_{i,},\ T,\ H$, aby $T \oplus T_i = F_{K'}(H) \oplus F_{K'}(H_i)$. Opět musí bez možnosti dotazovat se orákula předpovědět pro orákulum $F_{K'}(\cdot)$ hodnotu $F_{K'}(H) \oplus F_{K'}(H_i)$ pro vybraná $H,\ H_i$, nebo hodnotu $F_{K'}(H)$ pro vybrané H. To se ale dle definice AXU povede s pravděpodobností nanejvýš $\mathbf{Adv}_F^{\mathrm{axu}}(\widehat{\sigma} + \rho)$ a celý důkaz je tak u konce.

■

Kapitola 6

Závěrečné shrnutí důkazové techniky

Jak bylo slíbeno v úvodu, v závěrečné kapitole přinášíme celkové zhodnocení a pokus o shrnutí jednotlivých důkazů a technik jejich provedení. Kromě prvních dvou tvrzení druhé kapitoly[1] vždy sestavujeme nějaké schéma ze schémat již existujících, o kterých předpokládáme, že jsou bezpečná.

Důkazy vypadají tak, že na začátku předpokládáme, že máme libovolného útočníka, který úspěšně útočí na nějakou vlastnost nově vytvořeného schématu. Máme-li takového útočníka k dispozici, můžeme sestrojit jiného útočníka nebo útočníky, kteří budou útočit na původní schéma a daného útočníka využijí jako svůj podprogram. Nový útočník je vždy definován tak, aby dokonale simuloval prostředí, tedy především tvar odpovědí, které útočník-podprogram očekává. Ten proto provede útok úplně stejně, jako kdyby opravdu útočil na nové schéma. Z toho plyne, že můžeme hodnotu jeho advantage považovat za nezávislou na novém útočníkovi. Po ukončení běhu vydá útočník-podprogram nějaký výstup. Nový útočník pak dle povahy útoku tento výstup buď převezme, nebo pomocí něho vytvoří vlastní výstup.

Obecně v těchto důkazech nastávají dva případy. V prvním z nich stačí sestrojit pouze jednoho nového útočníka, pro kterého platí, že vždy, když jeho útočník-podprogram provede svůj simulovaný útok úspěšně, bude úspěšný celý útok. Z toho potom vyplývá, že advantage nového útočníka musí být alespoň tak velká, jako je advantage útočníka-podprogramu. To by ovšem znamenalo, že máme útočníka, který úspěšně útočí

[1] Oba důkazy probíhají v rámci jediného schématu. První důkaz je triviální a vůbec ho nebudeme uvažovat, o druhém jen řekneme, že probíhá vlastně na stejném principu jako důkazy ostatní.

na původní schéma. Protože ale předpokládáme, že původní schéma je bezpečné, musí být tato hodnota velmi nízká, a tedy i nově vytvořené schéma je bezpečné. Ve druhém případě pak jediný nový útočník k důkazu nestačí a je nutné jich sestrojit několik. Postupuje se tak, že pravděpodobnost úspěchu útoku daného útočníka-podprogramu (tedy jeho advantage) se vyjádří pomocí součtu pravděpodobností, které lze shora omezit hodnotami advantage nových útočníků, případně nějakou zanedbatelnou konstantou. Jednotlivé nerovnosti se pak dokazují stejně jako v prvním případě. V případě omezení konstantou je platnost dokázána jiným způsobem. Získáváme tak, že hodnota součtu advantage nových útočníků, případně doplněná o zanedbatelnou konstantu, je alespoň tak veliká, jako je advantage útočníka-podprogramu. Protože původní schéma je bezpečné, je hodnota tohoto součtu velmi nízká, tedy je nízká i hodnota advantage útočníka-podprogramu a opět se ukázalo, že nové schéma je bezpečné.

Závěrem tedy pouze poznamenejme, že opravdu existuje obecná metoda dokazování bezpečnosti pro schémata symetrické kryptografie, a to taková, kterou jsme popsali výše. Jedinou nevýhodou tohoto přístupu nejspíše je, že musíme vycházet z předpokladu bezpečnosti základních stavebních prvků schématu, a tu je třeba dokázat jinými metodami. Na druhou stranu ale platí, že pokud by bezpečnost některého z těchto prvků byla v praxi prolomena, stačí jej pouze nahradit novým, bezpečným prvkem a bezpečnost celého schématu nebude nikterak narušena.

Literatura

[1] Bellare M., Namprempre Ch. (2000): *Authenticated Encryption: Relations among Notions and Analisys of the Generic Composition Paradigm.*
`http://cseweb.ucsd.edu/ mihir/papers/oem.pdf`

[2] Bellare M., Canetti R., Krywczyk H. (1996): *Keying Hash Function for Message Authentication.*
`http://cseweb.ucsd.edu/ mihir/papers/kmd5.pdf`

[3] Rogaway P. (2006): *Authenticated Encryption with Associated Data.* `http://www.cs.ucdavis.edu/ rogaway/papers/ad.pdf`