Pokročilá kryptologie

Formátování a doplnění zpráv

Ing. Josef Kokeš, prof. Ing. Róbert Lórencz, CSc.

České vysoké učení technické v Praze, Fakulta informačních technologií
Katedra informační bezpečnosti

Padding

Padding

- Padding („vycpávka“) je kryptografická technika, která umožňuje dosáhnout určitých vlastností prostřednictvím vhodných úprav otevřeného textu. Šifrovací algoritmus samotný se nemění, liší se zpracování otevřeného textu před vstupem do šifrovacího algoritmu a po výstupu z dešifrovacího algoritmu.
- Typická použití:
  - Maskování vlastností (délka, struktura) otevřeného textu, znesnadnění nebo znemožnění některých kryptoanalytických metod
  - Doplnění otevřeného textu na zadanou délku (operační režimy blokových šifer, hashovací funkce)
  - Ochrana otevřeného textu proti pozměnění (asymetrické šifry)
  - Ochrana integrity zprávy (MAC = message authentication code) proti některým útokům
- Pozor! Špatně použitý padding může vést až k získání otevřeného textu!

Maskování vlastností otevřeného textu (1)

Maskování vlastností otevřeného textu

- Historické pojetí: Mnohé zprávy vykazují standardní strukturu, kterou útočník zná nebo může odhadnout - dopisy začínají pozdravem a oslovením a končí podpisem odesilatele, tabulkové údaje (hodnota vybraných měn v jednotlivých dnech v měsíci) stále opakují konstantní množinu sloupečků, vysílání lodi začne její identifikací, časem a polohou.
- Padding spočívá v přeházení pořadí jednolivých části podle určitých pravidel (oslovení se ocitne někde uprostřed zprávy, kde ho skryjí předcházející texty) nebo v doplnění umělých symbolů bez vlastního významu (např. Caesarova šifra a doplnění znaků tak, aby všechny měly stejnou frekvenci výskytu).
- Tato technika se ale používá i dnes

Maskování vlastností otevřeného textu (2)

Maskování vlastností otevřeného textu - příklady

- Příklad: TOR (The Onion Ring) - anonymizační síť, její klient vytváří a odesílá nesmyslné zprávy na nesmyslné adresy a tím maskuje existenci zprávy (útočník nepozná, jestli probíhající komunikace je tvořena smysluplným obsahem nebo vycpávkou) i adresáta (stejnou zprávu lze odeslat deseti adresátům a útočník neví, kterému z nich byla skutečně určena).
- Další příklad: Protokoly internetové telefonie typicky používají kompresi. Ticho lze komprimovat lépe než hlas, různé kobminace hlásek mají různou typickou úroveň komprese. Podle objemu dat a střídání velikosti packetů může útočník odhadnout dobu, po kterou oběť mluví, případně dekódovat i vlastní obsah. Přidáním umělé informace je tato možnost velmi ztížena.

Doplnění otevřeného textu na zadanou délku (1)

Doplnění otevřeného textu na zadanou délku

- Blokové šifry i hashovací funkce pracují s bloky určité délky. Co když vstupní data mají délku, která není délkou bloku beze zbytku dělitelná?
- Řešení: Doplnit poslední blok dalšími daty.
- Naivní varianty:
  - Náhodná data: Stanou se pevnou součástí otevřeného textu, není jak poznat, kterou část bloku tvoří užitečná data a kterou padding (Např. `Cena = 1234$a` - je padding jen `a` a cena je 1234 USD, nebo je padding `$a` a cena je 1234 Kč? Není dokonce padding `4$a`?). Navíc uvažujme, že počítáme hash souboru - s náhodným paddingem se bude hash při každém výpočtu lišit, přestože obsah souboru se nezmění.

Doplnění otevřeného textu na zadanou délku (2)

- Naivní varianty (pokračování):
  - Samé nuly (nebo jiný pevný symbol): Jako výše - co když sám otevřený text obsahuje nuly? V případě hashe ztrácíme bezkoliznost - útočník může vygenerovat další zprávu se stejným hashem tím, že k původnímu otevřenému textu přidá nulu.

Pořadavky na funkční řešení

- Padding musíme umět zase odstranit
- Padding by měl být deterministický
- Padding nesmí dát útočníkovi informaci o otevřeném textu

Doplnění otevřeného textu na zadanou délku (3)

Bitový padding

- Předpokládáme blok délky n bitů, otevřený text OT délky m bitů, $m \leq n$.
- Bitový padding má tvar 1000...00, kde počet nul je $n - m - 1$
- Připojuje se za konec OT
- Odstranění: Postupujeme od konce OT, dokud nenarazíme na první jedničku. Všechno od ní až do konce odstraníme a dostaneme původní zprávu.
- Co když $n = m$? Pak odstraníme 1 nebo víc bitů zprávy!
- Standardní řešení (shodné i pro ostatní paddingová schémata): Přidáme další blok, který obsahuje pouze padding.

Doplnění otevřeného textu na zadanou délku (4)

Bajtový padding

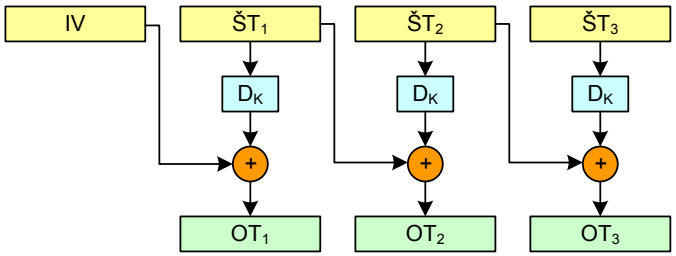
- Předpokládáme blok délky n bajtů, OT délky m bajtů, $m < n$.
- Mnoho různých schémat, například:
  - ANSI X.923: Padding je tvořen bajty `00 00 ... 00`[1] `x`, kde $x = n - m$ a počet 0 je roven $x - 1$. Odstranění spočívá v přečtení posledního bajtu a odstranění jím určeného množství bajtů z konce zprávy.
  - ISO 10126: Obdoba ANSI X.923, místo nulových bajtů jsou použity náhodné bajty. Pozor, toto schéma nesplňuje požadavek na determinističnost!
  - PKCS7: Padding je tvořen bajty `x x ... x`, kde $x = n - m$ a počet bajtů je roven x. Tzn. padding může být `01`, `02 02`, `03 03 03` atd. Odstranění je stejné jako u ANSI X.923.
  - ISO/IEC 7816-4: Obdoba bitového paddingu na úrovni bajtů; první bajt paddingu má hodnotu `80`, ostatní mají hodnotu `00`. Odstranění je totožné jako u bitového paddingu.

[1] Všechny hodnoty bajtů jsou hexadecimálně.

Padding Oracle Attack (1)

Útok typu Padding Oracle

- Mějme k dispozici počítač, který zpracovává zprávy šifrované nějakou blokovou šifrou v režimu CBC, který napřed testuje padding a teprve potom případně ověřuje integritu zprávy, a který umožňuje odlišit[2] chybu paddingu od chyby integrity (např. webový server používající starší verzi TLS protokolu). Tento počítač nám nechtěně umožní rozluštit cizí šifrované zprávy bez ohledu na použitou šifru.



- Připomínka: dešifrování CBC režimu:
- Útočník zná IV, ST_1, ST_2, ST_3, zná použité schéma paddingu (např. PKCS7) a chce zjistit OT_2.

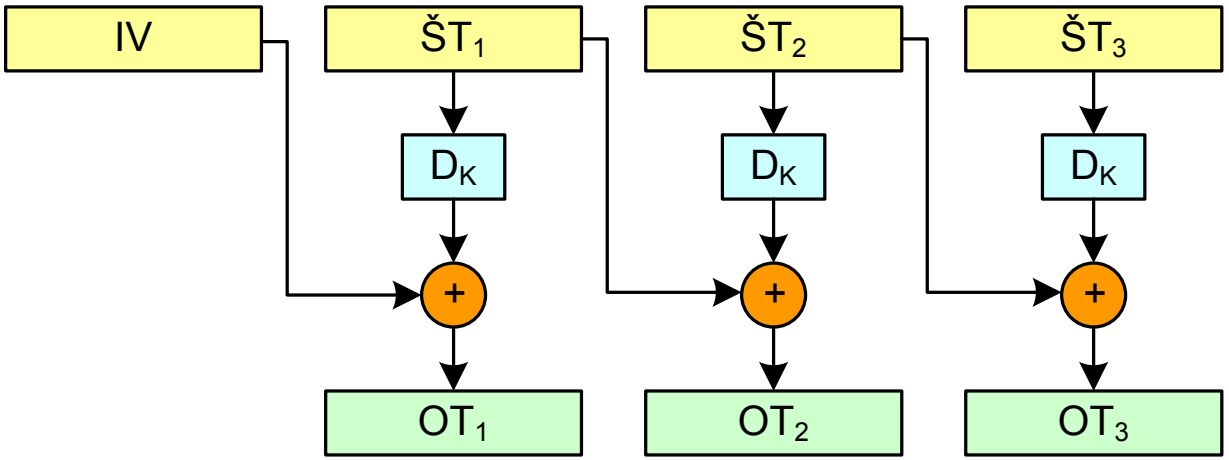
[2] Např. pomocí odlišných chybových hlášení nebo časovými závislostmi - chyba v 1. bajtu paddingu se projeví rychleji než chyba v 2. bajtu paddingu, chyba ve kterémkoliv bajtu paddingu se projeví rychleji než chyba ve vlastní zprávě.

Padding Oracle Attack (2)

- Útočník zná IV, ST_1, ST_2, ST_3 (odposlechnutá komunikace), zná použité schéma paddingu (např. PKCS7) a chce zjistit OT_2.
- Útočník si tipne, že hledaný poslední byte OT_2 (označme ho a_1) má hodnotu b_1.
- Přixoruje k poslednímu bajtu T_1 hodnotu $b_1 \oplus 01$, čímž se po dešifrování poslední byte OT_2 změní na $a_1 \oplus b_1 \oplus 01$.
- Pokud tedy $a_1 = b_1$, bude mít poslední bajt hodnotu `01` a bude platným jednobajtovým paddingem. Pošle tedy (IV, ST_1, ST_2) (ST_3 zahodil) serveru a sleduje, jakou dostane chybu:
- Pokud chybu paddingu, pak $a_1 \neq b_1$, změní tedy hodnotu b_1 a zkusí to znovu.
- Nejpozději po 256 pokusech dostane chybu vnitřku zprávy (server poznal, že zpráva není kompletní). V tu chvíli ví, že $a_1 = b_1$ a zná tedy správnou hodnotu a_1.

Padding Oracle Attack

Padding Oracle Attack (3)



- Útočník nyní zopakuje celý proces pro předposlední bajt OT_2 (a_2) a padding $(02, 02)$ (tzn. předposlední bajt xoruje $b_2 \oplus 02$, poslední bajt xoruje $a_1 \oplus 02$ a pošle serveru), atd. pro všechny bajty OT_2.
- Nyní může totéž zopakovat pro OT_1, přičemž serveru posílá jen (IV, ST_1).

Padding Oracle Attack (4)

Obrana proti Padding Oracle

- Upravit server tak, aby útočník nemohl rozlišit chybu paddingu od chyby zprávy. U časovacích útoků to může být značně složité kvůli optimalizujícím překladačům (překladač v dobré vůli ušetřit čas přeskočí „zbytečné“ průchody smyčkou).
- Upravit server tak, aby napřed otestoval integritu celé zprávy včetně paddingu a teprve potom případně padding odstraňoval (tzv. režim „encrypt-then-MAC“ místo „MAC-then-encrypt“). Pozn.: Nepomůže odstranit kontrolu integrity úplně, pokud útočník tále může rozpoznat, zda je padding v pořádku nebo ne.
- Použít šifru, která nepotřebuje padding, např. libovolnou proudovou šifru.
- Použít šifrovací režim, který nepotřebuje padding, např. režim čítače (CTR).

Padding u asymetrických šifer (1)

Padding u asymetrických šifer

- Asymetrická šifra se z pohledu paddingu chová obdobně jako bloková šifra, místo o délce bloku však hovoříme o délce modulu. Navíc ovšem:
- Asymetrické šifry jsou tzv. malleable, útočník může pozměnit otevřený text předvídatelným způsobem, aniž by musel šifru prolomit.
- Navíc pro RSA, bez použití paddingu může být prolomení některých šifrových textů triviální záležitostí.
- RSA bez paddingu je navíc deterministickou funkcí (stejná zpráva má vždy stejný šifrový text).

Padding u asymetrických šifer (2)

Malleability pro El Gamal

- Připomenutí. Buď x soukromý klíč, $(m, g, c = g^x)$ veřejný klíč, p otevřený text a y nonce zvolená odesilatelem. Pak $ST = (d = |g^y|_m, e = |c^y \cdot p|_m)$ je šifrový text šifry El Gamal. Dešifrování probíhá podle vztahu $c^y = |(g^x)^y|_m = |(g^y)^x|_m = |d^x|_m, C = |(c^y)^{-1}|_m, p' = |C \cdot e|_m = |(c^y)^{-1} \cdot c^y \cdot p|_m = p$.
- Útočník může vhodnou úpravou ST změnit p předvídatelným způsobem:
- $ST' = (d, s \cdot e) \Rightarrow p' = s \cdot p$
- Všimněte si, že útočník nemusí p znát, ale dokáže ho znásobit libovolnou konstantou s. Pokud p reprezentuje text, nejspíš se na změnu přijde, pokud by ale p reprezentovalo celé číslo (např. částku, číslo účtu, telefonní číslo...), nemusela by změna být detekována.

Padding u asymetrických šifer (3)

Malleability pro RSA

- Připomenutí. Buď d soukromý klíč, (n, e) veřejný klíč takový, že $|d \cdot e|_{\Phi}(n) = 1$, p otevřený text. Pak $ST = |p^e|_n$ je šifrový text šifry RSA. Dešifrování probíhá podle vztahu $|ST^d|_n = p$.
- Útočník může vhodnou úpravou ST změnit p předvídatelným způsobem:
- $ST' = s^e \cdot ST \Rightarrow p' = |ST'^d|n = |(s^e \cdot p^e)^d|m = s \cdot p$
- Opět, útočník dokázal předvídatelně změnit (vynásobit konstantou s) hodnotu p, aniž by ji znal.

Padding u asymetrických šifer (4)

Prolomení šifrového textu u RSA

- Mějme veřejný klíč ($2^{2047} < n < 2^{2048}, e = 3$)
- Nechť $p \approx 2^{512}$.
- Pak $ST = |p^e|_n \approx |2^{1536}|_n \approx 2^{1536}$.
- Útočník zná $e = 3$ (veřejný klíč), zkusí tedy spočítat $\sqrt[3]{ST}$ v reálné aritmetice, kde to je rychlé a snadné. A skutečně dostane $\sqrt[3]{p^3} = p$. Pro takto velký klíč a takto malý exponent triviálně získal 512 bitů (64 bajtů) otevřeného textu!

Padding u asymetrických šifer (5)

Determinističnost u RSA

- Šifrování probíhá podle vztahu $ST = |p^e|_n$. n je konstantní, e je konstantní, tudíž nutně $p_1 = p_2 \Leftrightarrow ST_1 = ST_2$
- Útočník pozná opakující se bloky obdobně jako v šifrovacím režimu ECB. Statistické vlastnosti těchto bloků mohou pomoci k odhalení otevřeného textu.

Padding u asymetrických šifer (6)

Jak pomůže padding: Malleability

- Padding se stává součástí p, tzn. $p = OT \odot PD$, kde OT je původní otevřený text, PD je padding a $\odot$ je operace spojení řetězců.
- Pak se ovšem padding stane i součástí šifrových textů:
- $ST_{ElGamal} = (d = |g^y|_m, e = |c^y \cdot p|_m) = (d = |g^y|_m, e = |c^y \cdot (OT \odot PD)|_m)$
- $ST_{RSA} = |p^e|_m = |(OT \odot PD)^e|_m$
- Pokud útočník změní hodnotu p (viz předchozí slajdy), změní jak hodnotu OT, tak hodnotu PD, a tuto změněnou hodnotu PD dokáže oběť detekovat, protože PD je tvořeno deterministicky.

Padding u asymetrických šifer (7)

Jak pomůže padding: Zjištění otevřeného textu RSA

- Nenulový padding zvyšuje bitovou délku p až o délku PD (v případě, že padding tvoříme např. podle schématu 1000...0001). Pak mocnění p i s malým šifrovacím exponentem e jistě způsobí překročení hranice n, uplatní se operace modulo a útočník ztratí možnost použít rychlou reálnou odmocninu.

Padding u asymetrických šifer (8)

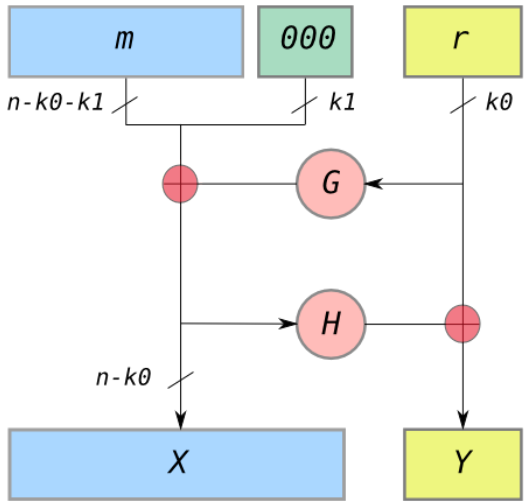
Jak pomůže padding: Determinističnost RSA

- Zvolíme-li padding tak, aby jeho část byla generována náhodně, pak při každém šifrování budeme mít jinou hodnotu PD a v důsledku toho i jinou hodnotu $p = OT \odot PD$. Pak také $ST = |p^e|_m$ bude mít odlišnou hodnotu.
- Je třeba zvážit, jak velkou část paddingu nastavíme náhodně: Hledáme kompromis mezi co nejdelší deterministickou částí (řeší malleabilitu šifry) a co nejdelší náhodnou částí (řeší determinističnost šifry).
- Vhodný mechanismus pro volbu paddingu, kombinující náhodnou a deterministickou část, je Optimal Asymmetric Encryption Padding. Padding pro RSA nalezněte v materiálech k přednášce v souboru `rsa-padding.pdf`.

Padding u asymetrických šifer (9)

Optimal Asymmetric Encryption Padding

- Mějme asymetrickou[3] blokovou šifru s blokem délky n bitů.
- Zvolíme konstanty k_0, k_1 tak, že $k_0 + k_1 < n$, a náhodná orakula $G : \{0,1\}^{k_0} \longrightarrow \{0,1\}^{n-k_0}$ a $H : \{0,1\}^{n-k_0} \longrightarrow \{0,1\}^{k_0}$. Těmito parametry jsme definovali následující paddingové schéma:



4

[3] Proč asymetrickou?

[4] Zdroj: OAEP diagram, Wikipedia Commons, http://en.wikipedia.org/wiki/File:Oaep-diagram-20080305.png

Padding u asymetrických šifer (10)

Optimal Asymmetric Encryption Padding

- Otevřený text rozdělíme na bloky m_i délky $n - k_0 - k_1$ (ne n jako u normálně použité asymetrické šifry). r_i je náhodně generovaný obsah. Spočítáme X_i a Y_i, jejich spojení $X_i \odot Y_i$ se stane vstupem (otevřeným textem) pro asymetrickou šifru.
- OAEP má strukturu Feistelovy sítě.
- Vlastnosti:
  - Náhodná část odstraňuje determinističnost šifrovací funkce.
  - Znemožňuje částečné dešifrování (pro získání m z X a Y musíme dešifrovat celé X a Y). Správnost odstranění paddingu poznáme podle toho, že v otevřeném textu nalezneme k_1 nulových bitů na správné pozici.
  - Místo k_1 nulových bitů můžeme použít i vhodný MAC (např. HMAC). Ovšem v režimu MAC-then-encrypt $\Rightarrow$ pozor na Padding Oracle!!

Ochrana integrity zprávy

- Viděli jsme, že padding může v sobě přímo zahrnovat MAC, jako v případě modifikovaného OAEP.
- Pokud paddingové schéma neumožňuje jednoznačné odstranění paddingu, může umožnit podvržení falešných zpráv se správným MAC.
- Nevhodně zvolený padding může znemožnit kontrolu integrity zpráv.
- Mějme funkci H pro vytvoření MAC, přijímající na vstupu zprávu o délce n bitů. Buď p zpráva o délce $m < n$ bitů.
- Zvolíme-li pro doplnění zprávy p tzv. zero padding (doplnění $n - m$ nulami, $p' = p \odot 000...000$), pak jistě platí:
  $H(p) = H(p \odot 0) = H(p \odot 00) = ...$
- Tzn. útočník může vygenerovat až $n - m$ jiných zpráv, které budou vyhovovat obětí spočítanému MAC.
- S vhodně zvoleným paddingem by tato situace nenastala: Např. pro binární padding platí, $p_1 \neq p_2 \Rightarrow (p_1 \odot PD_1) \neq (p_2 \odot PD_2)$ a tedy $H(p) \neq H(p \odot 0)$.