Šifrovací standard AES

Představujeme kandidáty na AES: Šifra RIJNDAEL

RIJNDAEL je jedním z pěti kandidátů na Advanced Encryption Standard (AES). O celém výběrovém řízení se podrobněji dozvíte v úvodu k této sérii stručných popisů všech finalistů, a to v článku „Bitva o trůn vrcholí" v Chipu 10/99; zde se už věnujeme přímo technickému popisu šifry. Připomeňme jen, že AES se stane šifrovacím standardem pro příští století (nebo alespoň pro nějaká ta desetiletí) a bude mít dalekosáhlý vliv na počítačovou bezpečnost.

Blokovou šifru ***RIJNDAEL*** přihlásili do soutěže známí kryptologové Joan Daemen a Vincent Rijmen. Ačkoliv jejich šifra podporuje i větší bloky, pro AES je délka vstupního a výstupního bloku definována jako 128 bitů. Délka klíče je volitelně 128, 192 a 256 bitů, což je **Nk** (= 4, 6 nebo 8) 32bitových slov.

RIJNDAEL je velmi flexibilní. I když jeho popis uvedeme v bajtech, lze jej elegantně zapsat i v 32bitových slovech. Návrh je přímočarý a za základ jsou použity operace v různých algebraických strukturách. Pracuje se s prvky *Galoisova tělesa* GF(2^8) a s polynomy, jejichž koeficienty jsou prvky z GF(2^8). Příslušné operace s nimi lze provádět buď tabulkově, nebo výpočtem přímo, což je v prvním případě výhodné pro implementaci softwarovou a v druhém případě pro hardwarovou. Bajtově orientovaný návrh také umožňuje optimalizovat programový kód pro různé mikroprocesory. Pro operace zašifrování a odšifrování sice není možné využít úplně totožný hardware (jako tomu bylo u šifry MARS), značnou část jeho prvků však použít lze.

Než přistoupíme k základním operacím, vysvětlíme si nejnutnější pojmy. Prvky v Galoisově tělese GF(2^8) mají osm bitů (b_7, ..., b_0), nereprezentují však bajty, nýbrž polynomy ($b_7x^7 + ... + b_1x^1 + b_0$). Násobení těchto prvků je proto zavedeno nikoli jako násobení bajtů, ale jako násobení jim odpovídajících polynomů, a to modulo $m(x) = x^8 + x^4 + x^3 + x^1 + 1$.

Takže například ´57´ (v apostrofech píšeme běžné hexadecimální vyjádření bitů b_7, ..., b_0) krát ´83´ je rovno ´C1´, neboť $(x^6 + x^4 + x^2 + x^1 + 1) * (x^7 + x^1 + 1) = (x^7 + x^6 + 1) \bmod m(x)$.

Postup při zašifrování

RIJNDAEL pracuje v rundách. Jejich počet **Nr** = 10, 12 a 14 je určen podle toho, jak dlouhý je šifrovací klíč, a odpovídá hodnotám **Nk** = 4, 6 a 8. Pro delší klíč se tedy použije více rund. Před operací zašifrování (nebo v jejím průběhu, tzv. „on-the-fly") se vypočítá 4 + Nr*4 rundovních klíčů (32bitových slov). První čtyři se „naxorují" na otevřený text (tzv. „whitening"). Potom proběhne **Nr** rund

Jedna runda zašifrování

```
Round (State, RoundKey)
{
ByteSub (State);
ShiftRow (State);
MixColumn (State); (neprovadi se v posledni runde)
AddRoundKey (State, RoundKey);
}
```

matice A

$$\begin{pmatrix} a_{00} & a_{01} & a_{02} & a_{03} \\ a_{10} & a_{11} & a_{12} & a_{13} \\ a_{20} & a_{21} & a_{22} & a_{23} \\ a_{30} & a_{31} & a_{32} & a_{33} \end{pmatrix}$$

ByteSub

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} = \begin{bmatrix} 1&0&0&0&1&1&1&1 \\ 1&1&0&0&0&1&1&1 \\ 1&1&1&0&0&0&1&1 \\ 1&1&1&1&0&0&0&1 \\ 1&1&1&1&1&0&0&0 \\ 0&1&1&1&1&1&0&0 \\ 0&0&1&1&1&1&1&0 \\ 0&0&0&1&1&1&1&1 \end{bmatrix} + \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \\ c_6 \\ c_7 \end{bmatrix}$$

MixColumn

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} 02&03&01&01 \\ 01&02&03&01 \\ 01&01&02&03 \\ 03&01&01&02 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

Obr. 1. Jedna runda zašifrování.

a v každé z nich se použijí 4 rundovní klíče. Na počátku se 16 bajtů otevřeného textu naplní postupně po sloupcích (tj. shora dolů a zleva doprava) do matice bajtů $\mathbf{A} = (a_{ij})_{i=0..3,\, j=0..3}$ a na ně se ve stejném pořadí postupně „naxoruje" 16 bajtů tvořících první čtyři rundovní klíče.

Poté proběhne **Nr** rund podle pseudokódu na obr. 1, kde „State" znamená stav matice **A**. Připomeňme, že prvky matice **A** jsou sice bajty, ale při násobení jsou chápány jako prvky GF(2^8). „Sčítání" těchto prvků (při operaci *MixColumn*) je běžná operace XOR. Výsledný šifrový text se opět vybírá po sloupcích z matice **A**.

Hlavní transformace

Všechny rundy jsou stejné, až na poslední, kde je malá změna – neprovádí se operace mixování MixColumn. Nyní k jednotlivým operacím z obrázku 1:

ByteSub je bajtová substituce (a→b), kterou aplikujeme na každý bajt $\mathbf{a}_{i,j}$ matice **A**. Nejprve vypočteme multiplikativní inverzi prvku **a**, tj. $\mathbf{c} = \mathbf{a}^{-1} \bmod m(x)$, a poté bajt **c** transformujeme na **b** substitucí **S** podle obr. 1. Substituci nemusíme počítat podle tohoto vzorce, ale můžeme si ji uložit jako pevnou tabulku.

ShiftRow vykoná v matici **A** cyklickou rotaci jejích prvků v jednotlivých řádcích doleva, a to tak, že první řádek ponechá beze změny, druhý rotuje o jednu pozici, třetí o dvě a čtvrtý o tři pozice.

MixColumn zesložití prvky v rámci každého sloupce matice **A**. Vstupem této transformace jsou všechny prvky daného sloupce (na obrázku je označen **a**) a výstupem jejich nové hodnoty (**b**). Tak bude například b_0 = ´02´*a_0 ⊕ ´03´*a_1 ⊕ ´01´*a_2 ⊕ ´01´*a_3.

Nakonec se operací *AddRoundKey* na prvky matice **A** (opět po sloupcích) „naxorují" po řadě jednotlivé bajty čtyř rundovních klíčů, které jsou na řadě. A to je celé.

Odšifrování probíhá trochu jinak než zašifrování, ale využívá jeho stavební prvky (popis je uveden v hlavním dokumentu popisujícím šifru; viz infotipy). Zbývá popsat výpočet rundovních klíčů ze šifrovacího klíče.

Zpracování klíče

Šifrovací klíč **key** (viz obr. 2) o **Nk** 32bitových slovech (4, 6 nebo 8) se naplní na počátek pomocného pole 32bitových slov **W**[0 ... Nk-1]. Toto pole se poté expanduje tak, že každé nové W je vypočítáno

jako W[i] = W[i - Nk] ⊕ temp, kde **temp** je W[i - 1] nebo jeho modifikace – viz obrázek 2. Při modifikaci se využívá operace cyklického posuvu bajtů slova **temp** o jeden doprava (*RotByte*), dále nám známé substituce bajtů *SubByte*, a to aplikované na každý bajt proměnné **temp**, a pole konstant **Const**[].

Implementace a rychlost

Dnešní implementace šifry RIJNDAEL v jazyce C na referenčním PC s Pentiem Pro 200MHz dosahují rychlosti šifrování cca 70/60/50 Mb/s při délkách klíče 128/192/256 bitů. Rychlost šifrování měřená počtem cyklů na jeden 128bitový blok je 363/432/500 cyklů (pro tytéž délky klíče); jde tedy zhruba o 3 – 5 cyklů na jeden bit. Na osmibitovém procesoru Intel 8051 trvá zašifrování jednoho bloku cca 3000 – 5000 cyklů (1 cyklus = 12 period oscilátoru) a na čipu Motorola 68HC08 (1 cyklus = 1 perioda oscilátoru) je to cca 8000 – 12 000 cyklů. Spotřeba paměti RAM je pouhých 52 bajtů (!), neboť u obou těchto implementací byly rundovní klíče počítány on-the-fly. Délka kódu je v obou případech do 1 KB. Odšifrování trvá vždy cca o 30 % déle než zašifrování.

Expanze klíče

```
for i = Nk to 4*Nr + 3 do
{
  temp = W[i - 1];
  if (i mod Nk = 0)
    temp = SubByte(RotByte(temp)) ⊕ Const[i / Nk];
  if ((i mod Nk = 4) AND (Nk = 8))
    temp = SubByte(temp);
  W[i] = W[i - Nk] ⊕ temp;
}
```

Obr. 2. Expanze klíče.

Bezpečnost

Oba autoři dokazují skvělé vlastnosti stavebních bloků schématu i odolnost vůči lineární a diferenciální kryptoanalýze. Protože schéma pro zašifrování i odšifrování (v hardwaru) se liší, není tu riziko slabých klíčů. Ekvivalenci klíčů (což je případ, kdy různé šifrovací klíče dávají stejné sady rundovních klíčů) brání podle autorů nelineární expanze.

infotipy

Zdrojové kódy:
ftp://ftp.funet.fi/pub/crypt/cryptography/symmetric/rijndael/
Úplný popis:
http://csrc.nist.gov/encryption/aes/aes_home.htm

Závěr

U šifry RIJNDAEL je ceněn její průzračný návrh, založený na různých algebraických operacích. Šifra je flexibilní při realizaci na různých typech procesorů s velmi malými nároky na paměť i velikost kódu, a přitom vykazuje ještě dostatečnou rychlost. Je vhodná i pro paralelní zpracování a je odolná vůči fyzickým typům útoků. Z mého pohledu jsou však navržené stavební prvky i jejich kompozice poměrně nové a osobně bych byl překvapen, kdyby RIJNDAEL zvítězil.

VLASTIMIL KLÍMA (*VKLIMA@DECROS.CZ*)