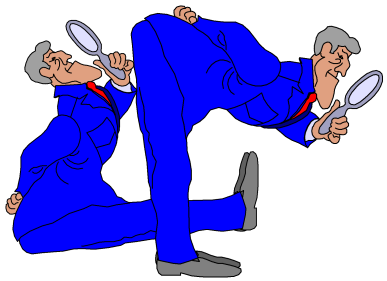
4. One-time Pad a lineární posouvací registry

Jan Paseka

Ústav matematiky a statistiky
Masarykova univerzita

23. listopadu 2023

O čem to bude



Popis systému I

FI Nyní budeme hovořit o následujícím perfektním systému: Předpokládejme, že abeceda Σ je obvyklá 26-ti písmenná anglická abeceda a že tečky, mezery atd. jsou vypuštěny a že odesílaná zpráva M sestává z N písmen. Abychom zašifrovali zprávu, vygenerujeme náhodnou posloupnost o N písmenech z abecedy Σ, přičemž výběr každého písmene je nezávislý a každé písmeno má pravděpodobnost $\frac{1}{26}$, že bude vybráno.

Popis systému I

FI Nyní budeme hovořit o následujícím perfektním systému: Předpokládejme, že abeceda Σ je obvyklá 26-ti písmenná anglická abeceda a že tečky, mezery atd. jsou vypuštěny a že odesílaná zpráva M sestává z N písmen. Abychom zašifrovali zprávu, vygenerujeme náhodnou posloupnost o N písmenech z abecedy Σ, přičemž výběr každého písmene je nezávislý a každé písmeno má pravděpodobnost $\frac{1}{26}$, že bude vybráno.

Tato náhodná posloupnost $(Z_1, \ldots, Z_N)$ bude klíč K a, abychom zašifrovali $M = (x_1, \ldots, x_N)$ pomocí K, budeme definovat $C = e(M, K)$ jako

$$y_i = x_i \oplus Z_i \bmod 26,$$

kde jako v obvyklém substitučním číslicovém systému jsme písmenům po řadě přiřadili číselnou hodnotu z množiny $\{0, \ldots, 25\}$.

Popis systému II

Tedy jako klíče vybereme rovněž všech 26^N posloupností délky N; každou z těchto posloupností lze zvolit se ***stejnou pravděpodobností*** tj.

$$H(K) = N\log 26.$$

Popis systému II

Tedy jako klíče vybereme rovněž všech 26^N posloupností délky N; každou z těchto posloupností lze zvolit se ***stejnou pravděpodobností*** tj.

$$H(K) = N\log 26.$$

Protože klíč $K = (Z_1, Z_2, \ldots, Z_N)$ je posloupnost náhodných písmen z 26-ti prvkové abecedy Σ, je zřejmé, že existuje 26^N stejně pravděpodobných kryptogramů. Zároveň platí

$$P(K|C) = \frac{1}{26^N}$$

pro všechna $K \in \mathbf{K}$.

Popis systému II

Tedy jako klíče vybereme rovněž všech 26^N posloupností délky N; každou z těchto posloupností lze zvolit se ***stejnou pravděpodobností*** tj.

$$H(K) = N\log 26.$$

Protože klíč $K = (Z_1, Z_2, \ldots, Z_N)$ je posloupnost náhodných písmen z 26-ti prvkové abecedy Σ, je zřejmé, že existuje 26^N stejně pravděpodobných kryptogramů. Zároveň platí

$$P(K|C) = \frac{1}{26^N}$$

pro všechna $K \in \mathbf{K}$.
Z kritéria 3 vidíme, že tento tzv. one-time pad systém $\langle \mathbf{M}, \mathbf{K}, \mathbf{C} \rangle$ je ***perfektní***, neboť klíč je jednoznačně určen zprávou a kryptogramem, množina zpráv je je zároveň množinou klíčů, resp. kryptogramů, zejména tedy mají stejnou mohutnost.

Popis systému II

Tedy jako klíče vybereme rovněž všech 26^N posloupností délky N; každou z těchto posloupností lze zvolit se ***stejnou pravděpodobností*** tj.

$$H(K) = N\log 26.$$

Protože klíč $K = (Z_1, Z_2, \ldots, Z_N)$ je posloupnost náhodných písmen z 26-ti prvkové abecedy Σ, je zřejmé, že existuje 26^N stejně pravděpodobných kryptogramů. Zároveň platí

$$P(K|C) = \frac{1}{26^N}$$

pro všechna $K \in \mathbf{K}$.
Z kritéria 3 vidíme, že tento tzv. one-time pad systém $\langle \mathbf{M}, \mathbf{K}, \mathbf{C} \rangle$ je ***perfektní***, neboť klíč je jednoznačně určen zprávou a kryptogramem, množina zpráv je je zároveň množinou klíčů, resp. kryptogramů, zejména tedy mají stejnou mohutnost.

Popis systému III

Tento šifrovací systém byl objeven v roce 1926 americkým inženýrem společnosti AT&T Gilbertem S. **Vernamem** (v dřívějších dobách byla písmena klíče napsána na listech trhacího bloku a jakmile bylo klíčové písmeno použito, byl odpovídající list vytrhnut a zničen).

Popis systému III

Tento šifrovací systém byl objeven v roce 1926 americkým inženýrem společnosti AT&T Gilbertem S. **Vernamem** (v dřívějších dobách byla písmena klíče napsána na listech trhacího bloku a jakmile bylo klíčové písmeno použito, byl odpovídající list vytrhnut a zničen).

Dnes se one-time pad neprovozuje s písmeny nýbrž s bity. Pak $\mathbf{a}_i, \mathbf{k}_i \in \{0, 1\}$ a kryptogram $\mathbf{a}_1 \oplus \mathbf{k}_1 \mathbf{a}_2 \oplus \mathbf{k}_2 \ldots \mathbf{a}_n \oplus \mathbf{k}_n$ získáme pomocí binárního sčítání.

Popis systému III

Tento šifrovací systém byl objeven v roce 1926 americkým inženýrem společnosti AT&T Gilbertem S. **Vernamem** (v dřívějších dobách byla písmena klíče napsána na listech trhacího bloku a jakmile bylo klíčové písmeno použito, byl odpovídající list vytrhnut a zničen).

Dnes se one-time pad neprovozuje s písmeny nýbrž s bity. Pak $\mathbf{a}_i, \mathbf{k}_i \in \{0, 1\}$ a kryptogram $\mathbf{a}_1 \oplus \mathbf{k}_1 \mathbf{a}_2 \oplus \mathbf{k}_2 \ldots \mathbf{a}_n \oplus \mathbf{k}_n$ získáme pomocí binárního sčítání.

Pro bezpečnost tohoto systému je podstatné, že všechny posloupnosti délky n se vyskytují s toutéž pravděpodobností. Jinak řečeno: ***Bity klíčového slova musí být voleny náhodně.***

Popis systému IV

Nejlépe si to představíme tím způsobem, že vrháme ideální minci. V praxi používáme fyzikální náhodný zdroj a tím automaticky vytvoříme bity.

Popis systému IV

Nejlépe si to představíme tím způsobem, že vrháme ideální minci. V praxi používáme fyzikální náhodný zdroj a tím automaticky vytvoříme bity.

Za tuto formu perfektní bezpečnosti musíme – nikoliv neočekávaně – platit vysokou cenu. Pro tradiční one-time pad potřebujeme velké množství papíru, který musí být před útočníkem absolutně bezpečně ukryt. Proto se takovéto systémy používají jen zřídka.

Popis systému IV

Nejlépe si to představíme tím způsobem, že vrháme ideální minci. V praxi používáme fyzikální náhodný zdroj a tím automaticky vytvoříme bity.

Za tuto formu perfektní bezpečnosti musíme – nikoliv neočekávaně – platit vysokou cenu. Pro tradiční one-time pad potřebujeme velké množství papíru, který musí být před útočníkem absolutně bezpečně ukryt. Proto se takovéto systémy používají jen zřídka.

V druhé světové válce se one-time pad používal anglickou dešifrovací skupinou, aby zprostředkovala zprávy premiérovi, které byly Němci zašifrovány pomocí Enigmy a které Angličané rozšifrovali. Tímto způsobem si spojenci zajistili, že Němci až do konce války nevěděli, že Enigma byla rozluštěna.

Popis systému V

Jednou z dalších nevýhod tohoto systému je neexistence matematického způsobu generování nezávislých náhodných proměnných, které slouží jako klíč.

Popis systému V

Jednou z dalších nevýhod tohoto systému je neexistence matematického způsobu generování nezávislých náhodných proměnných, které slouží jako klíč.

Tedy je nutné použít pseudonáhodných posloupností generovaných jednou z mnoha standardních metod.

Popis systému V

Jednou z dalších nevýhod tohoto systému je neexistence matematického způsobu generování nezávislých náhodných proměnných, které slouží jako klíč.

Tedy je nutné použít pseudonáhodných posloupností generovaných jednou z mnoha standardních metod.

Neexistuje pak žádná záruka, že takováto pseudonáhodné posloupnosti nám budou stejnou úroveň bezpečnosti. Jedná se o hluboký matematický problém.

Popis systému V

Jednou z dalších nevýhod tohoto systému je neexistence matematického způsobu generování nezávislých náhodných proměnných, které slouží jako klíč.

Tedy je nutné použít pseudonáhodných posloupností generovaných jednou z mnoha standardních metod.

Neexistuje pak žádná záruka, že takováto pseudonáhodné posloupnosti nám budou stejnou úroveň bezpečnosti. Jedná se o hluboký matematický problém.

Proč se tento nepochybně perfektní systém používá jen velmi zřídka? Abychom byli schopni zodpovědět tuto otázku, představme si sebe v roli příjemce.

Použití systému I

Příjemce může přirozeně kryptogram pohodlně rozšifrovat: dešifrování je v podstatě stejný postup jako zašifrování (používáme-li bity, jedná se dokonce o přesně totéž). To ale může příjemce provést jen v případě, že má klíč.

Použití systému I

Příjemce může přirozeně kryptogram pohodlně rozšifrovat: dešifrování je v podstatě stejný postup jako zašifrování (používáme-li bity, jedná se dokonce o přesně totéž). To ale může příjemce provést jen v případě, že má klíč.

Kde je vlastně problém? Problém spočívá v tom, že musíme přenést (doručit) dlouhý tajný klíč.

Použití systému I

Příjemce může přirozeně kryptogram pohodlně rozšifrovat: dešifrování je v podstatě stejný postup jako zašifrování (používáme-li bity, jedná se dokonce o přesně totéž). To ale může příjemce provést jen v případě, že má klíč.

Kde je vlastně problém? Problém spočívá v tom, že musíme přenést (doručit) dlouhý tajný klíč.

Kdybychom toto prováděli pomocí stejné cesty jako zprávu, je vzhledem k délce klíče šance přečtení klíče stejná jako při předání nezašifrovaného textu zprávy.

Použití systému I

Příjemce může přirozeně kryptogram pohodlně rozšifrovat: dešifrování je v podstatě stejný postup jako zašifrování (používáme-li bity, jedná se dokonce o přesně totéž). To ale může příjemce provést jen v případě, že má klíč.

Kde je vlastně problém? Problém spočívá v tom, že musíme přenést (doručit) dlouhý tajný klíč.

Kdybychom toto prováděli pomocí stejné cesty jako zprávu, je vzhledem k délce klíče šance přečtení klíče stejná jako při předání nezašifrovaného textu zprávy.

Člověk by si mohl myslet, že u takovéhoto systému by mohl odesilatel zprávu nepříteli předat přímo do domu.

Použití systému II

To ale není zcela správné; totiž pro přenos klíče může odesilatel určit druh, způsob a dobu předání, což samozřejmě u přenosu zprávy neplatí. Jiný způsob přenosu klíče je použití kurýra.

Použití systému II

To ale není zcela správné; totiž pro přenos klíče může odesilatel určit druh, způsob a dobu předání, což samozřejmě u přenosu zprávy neplatí. Jiný způsob přenosu klíče je použití kurýra.

Při přenosu klíče se nejedná pouze o teoretický problém, nýbrž o to, že obtížnost výměny klíče silně ovlivňuje nasazení šifrovacích systémů.

Použití systému II

To ale není zcela správné; totiž pro přenos klíče může odesilatel určit druh, způsob a dobu předání, což samozřejmě u přenosu zprávy neplatí. Jiný způsob přenosu klíče je použití kurýra.

Při přenosu klíče se nejedná pouze o teoretický problém, nýbrž o to, že obtížnost výměny klíče silně ovlivňuje nasazení šifrovacích systémů.

Důležitý postup pro vyřešení tohoto problému spočívá v tom, že namísto ***skutečně náhodných klíčových posloupností*** použijeme pouze **pseudonáhodné** posloupnosti.

Použití systému II

To ale není zcela správné; totiž pro přenos klíče může odesilatel určit druh, způsob a dobu předání, což samozřejmě u přenosu zprávy neplatí. Jiný způsob přenosu klíče je použití kurýra.

Při přenosu klíče se nejedná pouze o teoretický problém, nýbrž o to, že obtížnost výměny klíče silně ovlivňuje nasazení šifrovacích systémů.

Důležitý postup pro vyřešení tohoto problému spočívá v tom, že namísto ***skutečně náhodných klíčových posloupností*** použijeme pouze **pseudonáhodné** posloupnosti.

Takováto posloupnost vypadá na první pohled jako skutečná náhodná posloupnost.

Použití systému III

A co je ještě důležitější: náhodnou posloupnost lze určit pomocí několika málo dat; tato data pak představují skutečný klíč.

Použití systému III

A co je ještě důležitější: náhodnou posloupnost lze určit pomocí několika málo dat; tato data pak představují skutečný klíč.

Oba komunikující partneři pak mohou z těchto dat spočítat náhodné posloupnosti a zašifrovat zprávu resp. dešifrovat kryptogram. Problém přenosu klíče tak není zcela vyřešen, ale podstatně ulehčen.

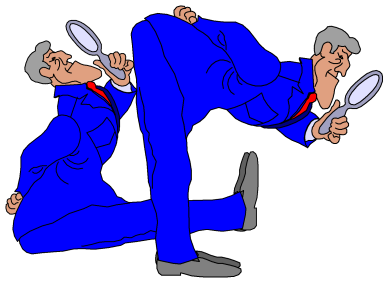
Použití systému III

A co je ještě důležitější: náhodnou posloupnost lze určit pomocí několika málo dat; tato data pak představují skutečný klíč.

Oba komunikující partneři pak mohou z těchto dat spočítat náhodné posloupnosti a zašifrovat zprávu resp. dešifrovat kryptogram. Problém přenosu klíče tak není zcela vyřešen, ale podstatně ulehčen.

Samozřejmě musíme za tuto výhodu zaplatit: takovéto systémy neposkytují žádnou perfektní bezpečnost. Budeme tedy hledat kompromis mezi docílenou bezpečností a množinou tajně přenositelných dat.

O čem to bude



1. One-time Pad
2. Posouvací registr
   - Generování
   - Vlastnosti posloupností
   - vytvořených lineárními posouvacími registry
   - Primitivní polynomy
3. Kryptoanalýza lineárních posouvacích registrů

Generování I

Posouvací registr je posloupnost v řadě za sebou následujících registrů, přičemž každý registr může obsahovat pouze číslici 1 (on) nebo 0 (off). Hodinový strojek reguluje chování systému, který pracuje v souladu s následujícími podmínkami:

Generování I

Posouvací registr je posloupnost v řadě za sebou následujících registrů, přičemž každý registr může obsahovat pouze číslici 1 (on) nebo 0 (off). Hodinový strojek reguluje chování systému, který pracuje v souladu s následujícími podmínkami:

Předpokládejme, že systém má m registrů $R_0, R_1, \ldots, R_{m-1}$ a že $X_i(t)$ označuje obsah registru R_i v čase t. Nechť je dále na začátku systém ve stavu

Generování I

Posouvací registr je posloupnost v řadě za sebou následujících registrů, přičemž každý registr může obsahovat pouze číslici 1 (on) nebo 0 (off). Hodinový strojek reguluje chování systému, který pracuje v souladu s následujícími podmínkami:

Předpokládejme, že systém má m registrů $R_0, R_1, \ldots, R_{m-1}$ a že $X_i(t)$ označuje obsah registru R_i v čase t. Nechť je dále na začátku systém ve stavu $\mathbf{X}(0) = (X_{m-1}(0), \ldots, X_0(0))$.

Generování I

Posouvací registr je posloupnost v řadě za sebou následujících registrů, přičemž každý registr může obsahovat pouze číslici 1 (on) nebo 0 (off). Hodinový strojek reguluje chování systému, který pracuje v souladu s následujícími podmínkami:

Předpokládejme, že systém má m registrů $R_0, R_1, \ldots, R_{m-1}$ a že $X_i(t)$ označuje obsah registru R_i v čase t. Nechť je dále na začátku systém ve stavu $\mathbf{X}(0) = (X_{m-1}(0), \ldots, X_0(0))$.

Pokud

$$\mathbf{X}(t) = (X_{m-1}(t), \ldots, X_0(t)).$$

označuje stav systému v době t, stav v čase $t+1$ je určen vztahy

$$X_i(t+1) = X_{i+1}(t) \qquad (0 \leq i \leq m-2), \tag{2.1}$$

$$X_{m-1}(t+1) = f(\mathbf{X}(t)), \tag{2.2}$$

kde f je nějaká binární funkce m proměnných.

Generování II

Pokud je f tvaru

$$f = \sum_{i=0}^{m-1} c_{m-i} X_i(t) = c_m X_0(t) \oplus c_{m-1} X_1(t) \oplus \cdots \oplus c_1 X_{m-1}(t),$$

mluvíme o ***lineárním posouvacím registru***.

Generování II

Pokud je f tvaru

$$f = \sum_{i=0}^{m-1} c_{m-i} X_i(t) = c_m X_0(t) \oplus c_{m-1} X_1(t) \oplus \cdots \oplus c_1 X_{m-1}(t),$$

mluvíme o ***lineárním posouvacím registru***.

Přitom chování systému je jednoznačně určeno

(a) počátečním stavem $\mathbf{X}(0)$ a

(b) množinou konstant $c_1, \ldots, c_m$.

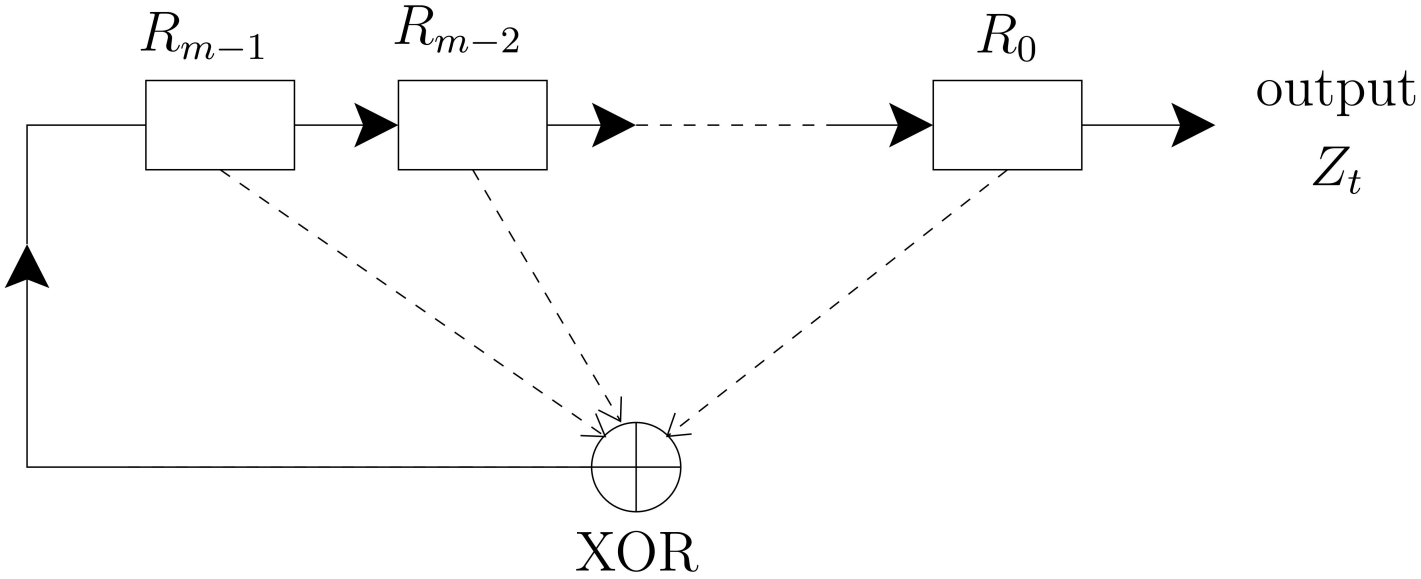
Budeme vždy předpokládat, že $c_m \neq 0$; jinak bychom mohli pracovat bez registru R_0.

Generování III

Způsob, jakým lineární systém pracuje, je, že po obdržení signálu každý registr provede dvě věci:

(i) Přenese svůj obsah do svého pravého souseda (registr R_0 toto provést nemůže, jeho obsah se stane výstupním bitem Z_t našeho stroje).

(ii) Takové registry R_i, pro které je $c_i = 1$ přenesou svůj obsah do čítače, ten je sečte a výsledek přenese do registru R_{m-1}.



Generování IV

Jakmile je jednou nastaven počáteční vektor, lze posouvací registr považovat za zdroj nekonečné posloupnosti binárních číslic

$$X_0(0), X_0(1), X_0(2), \ldots.$$

Generování IV

Jakmile je jednou nastaven počáteční vektor, lze posouvací registr považovat za zdroj nekonečné posloupnosti binárních číslic

$$X_0(0), X_0(1), X_0(2), \ldots.$$

Ačkoliv takto vytvořená posloupnost není náhodná, lze ukázat, že má jisté rysy nahodilosti. Navíc ji lze snadno a rychle generovat. Bohužel je však velmi nejistá.

Vlastnosti vytvořených posloupností I

Nejprve uvažujme periodicitu. Nekonečná posloupnost $(y_i : 0 \leq i < \infty)$ se nazývá ***periodická*** s ***periodou*** p, jestliže je p kladné přirozené číslo takové, že $y_{i+p} = y_i$ pro všechna i a navíc je p nejmenší kladné přirozené číslo s touto vlastností.

Vlastnosti vytvořených posloupností I

Nejprve uvažujme periodicitu. Nekonečná posloupnost $(y_i : 0 \leq i < \infty)$ se nazývá ***periodická*** s ***periodou*** p, jestliže je p kladné přirozené číslo takové, že $y_{i+p} = y_i$ pro všechna i a navíc je p nejmenší kladné přirozené číslo s touto vlastností.

Má-li tedy posloupnost $(y_i : 0 \leq i < \infty)$ periodu p, můžeme ji psát ve tvaru

$$y_0, y_1, y_2, \ldots, y_{p-1}, \quad y_0, y_1, y_2, \ldots, y_{p-1}, \ldots$$

Jinak řečeno, posloupnost s periodou p je přesně posloupnost opakování konečného bloku délky p.

Vlastnosti vytvořených posloupností II

Vraťme se nyní k posloupnosti určené lineárním posouvacím registrem: předpokládejme, že počáteční vektor $\mathbf{X}(0)$ není nulový vektor a že rovnice 2.1 a 2.2 lze přepsat ve tvaru

$$\mathbf{X}(t+1) = \mathbf{C}X(t), \tag{2.3}$$

kde $\mathbf{C}$ je matice tvaru

$$\mathbf{C} = \begin{bmatrix} c_1 & c_2 & c_3 & \dots & c_{m-1} & c_m \\ 1 & 0 & 0 & \dots & 0 & 0 \\ 0 & 1 & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 1 & 0 \end{bmatrix}.$$

Vlastnosti vytvořených posloupností III

Protože jsme předpokládali, že $c_m = 1$ a protože

$$\det \mathbf{C} = c_m = 1,$$

vidíme, že $\mathbf{C}$ je regulární matice. Iterováním rovnice 2.3 dostaneme $\mathbf{X}(t) = \mathbf{C}^t\mathbf{X}(0)$.

Vlastnosti vytvořených posloupností III

Protože jsme předpokládali, že $c_m = 1$ a protože

$$\det \mathbf{C} = c_m = 1,$$

vidíme, že $\mathbf{C}$ je regulární matice. Iterováním rovnice 2.3 dostaneme $\mathbf{X}(t) = \mathbf{C}^t\mathbf{X}(0)$.

Přitom platí

Věta 2.1

Posloupnost vytvořená pomocí lineárního posouvacího registru je periodická a pokud je vytvořena z m registrů, je její maximální perioda $2^m - 1$.

Vlastnosti vytvořených posloupností IV

Důkaz Věty 2.1.

Protože je $\mathbf{C}$ regulární, je i regulární matice $\mathbf{C}^i$ $(i = 0, 1, \dots)$; přitom je $\mathbf{X}(0)$ nenulový vektor a existuje právě $2^m - 1$ nenulových vektorů délky m.

Vlastnosti vytvořených posloupností IV

Důkaz Věty 2.1.

Protože je $\mathbf{C}$ regulární, je i regulární matice $\mathbf{C}^i$ $(i = 0, 1, \dots)$; přitom je $\mathbf{X}(0)$ nenulový vektor a existuje právě $2^m - 1$ nenulových vektorů délky m.

Je-li $k = 2^m - 1$, pak jsou

$$\mathbf{X}(0), \mathbf{C}\mathbf{X}(0), \mathbf{C}^2\mathbf{X}(0), \dots, \mathbf{C}^k\mathbf{X}(0)$$

nenulové vektory délky m a tudíž nemohou být všechny různé:

Vlastnosti vytvořených posloupností IV

Důkaz Věty 2.1.

Protože je $\mathbf{C}$ regulární, je i regulární matice $\mathbf{C}^i$ $(i = 0, 1, \dots)$; přitom je $\mathbf{X}(0)$ nenulový vektor a existuje právě $2^m - 1$ nenulových vektorů délky m.

Je-li $k = 2^m - 1$, pak jsou

$$\mathbf{X}(0), \mathbf{C}\mathbf{X}(0), \mathbf{C}^2\mathbf{X}(0), \dots, \mathbf{C}^k\mathbf{X}(0)$$

nenulové vektory délky m a tudíž nemohou být všechny různé: řekněme, že

$$\mathbf{C}^s\mathbf{X}(0) = \mathbf{C}^{s+t}\mathbf{X}(0),$$

kde $0 \leq s < s + t \leq 2^m - 1$.

Vlastnosti vytvořených posloupností V

Důkaz Věty 2.1 - pokračování.

Protože existuje $\mathbf{C}^{-\mathbf{s}}$, máme

$$\mathbf{X}(t) = \mathbf{C}^t\mathbf{X}(0) = \mathbf{C}^{-s}\mathbf{C}^{s+t}\mathbf{X}(0) = \mathbf{X}(0).$$

Vlastnosti vytvořených posloupností V

Důkaz Věty 2.1 - pokračování.

Protože existuje $\mathbf{C}^{-\mathbf{s}}$, máme

$$\mathbf{X}(t) = \mathbf{C}^t\mathbf{X}(0) = \mathbf{C}^{-s}\mathbf{C}^{s+t}\mathbf{X}(0) = \mathbf{X}(0).$$

Tedy

$$\mathbf{X}(r+t) = \mathbf{C}^{r+t}\mathbf{X}(0) = \mathbf{C}^r\mathbf{C}^t\mathbf{X}(0) = \mathbf{C}^r\mathbf{X}(t) = \mathbf{C}^r\mathbf{X}(0) = \mathbf{X}(r)$$

pro všechna $r \geq 0$, a $\mathbf{C}^t\mathbf{X}(0)$ je periodická s periodou nejvýše $t \leq 2^m - 1$.

Vlastnosti vytvořených posloupností V

Důkaz Věty 2.1 - pokračování.

Protože existuje $\mathbf{C}^{-\mathbf{s}}$, máme

$$\mathbf{X}(t) = \mathbf{C}^t\mathbf{X}(0) = \mathbf{C}^{-s}\mathbf{C}^{s+t}\mathbf{X}(0) = \mathbf{X}(0).$$

Tedy

$$\mathbf{X}(r+t) = \mathbf{C}^{r+t}\mathbf{X}(0) = \mathbf{C}^r\mathbf{C}^t\mathbf{X}(0) = \mathbf{C}^r\mathbf{X}(t) = \mathbf{C}^r\mathbf{X}(0) = \mathbf{X}(r)$$

pro všechna $r \geq 0$, a $\mathbf{C}^t\mathbf{X}(0)$ je periodická s periodou nejvýše $t \leq 2^m - 1$.

Posloupnost vytvořená pomocí lineárního posouvacího registru je tedy periodická.

∎

Primitivní polynomy I

Definujme ***charakteristický polynom*** lineárního posouvacího registru jako polynom

$$P_m(x) = 1 + \sum_{i=1}^{m} c_i x^i,$$

s $c_m \neq 0, c_i \in \{0, 1\}$.

Primitivní polynomy I

Definujme ***charakteristický polynom*** lineárního posouvacího registru jako polynom

$$P_m(x) = 1 + \sum_{i=1}^{m} c_i x^i,$$

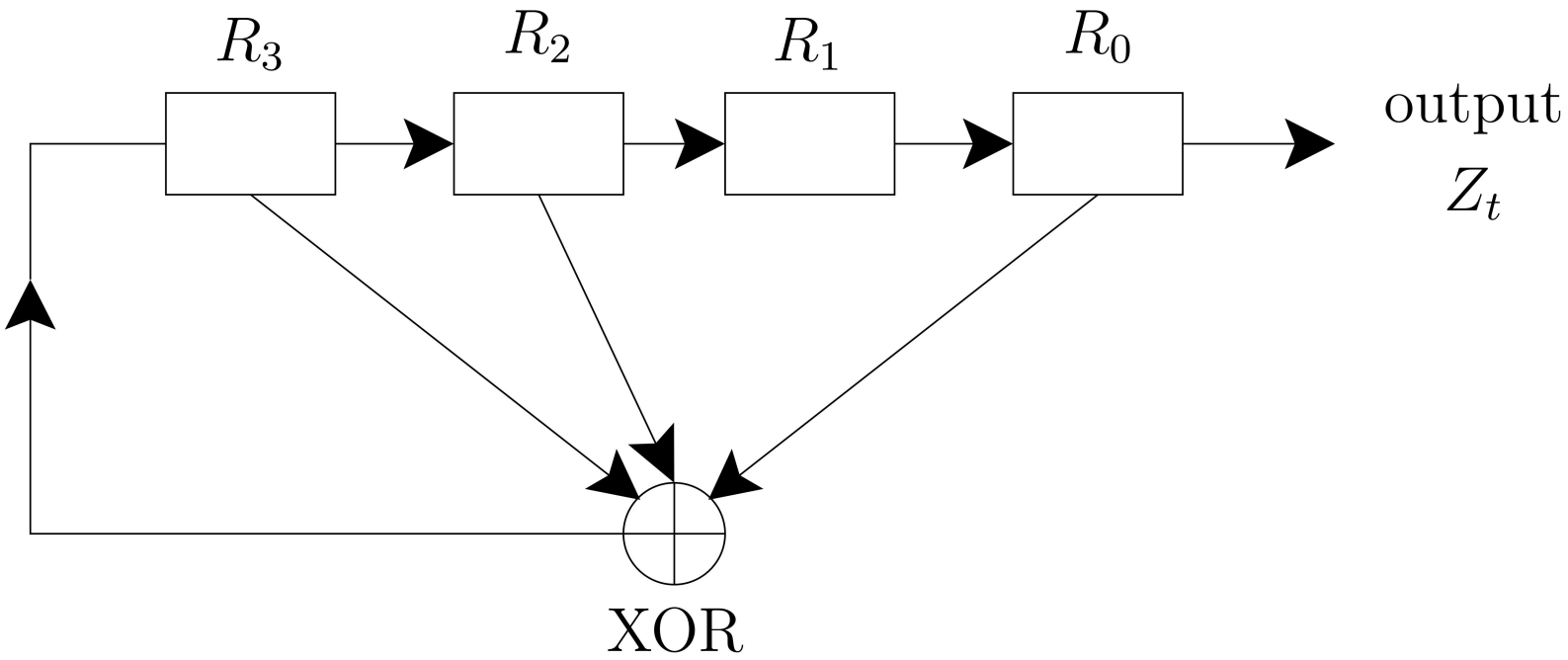
s $c_m \neq 0, c_i \in \{0, 1\}$.

Charakteristický polynom je ***primitivní***, jestliže

(a) nemá vlastní netriviální dělitele,

(b) $P_m(x)$ nedělí polynom $x^d + 1$ pro všechna $d < 2^m - 1$.

Primitivní polynomy II



Lineární posouvací registr s charakteristickým polynomem
$1 + x + x^2 + x^4$.

Primitivní polynomy III

Následující tvrzení uvedeme bez důkazu.

Věta 2.2

Posloupnost vytvořená pomocí lineárního posouvacího registru z nenulového vstupu má maximální periodu právě tehdy, je-li její charakteristický polynom primitivní.

Primitivní polynomy III

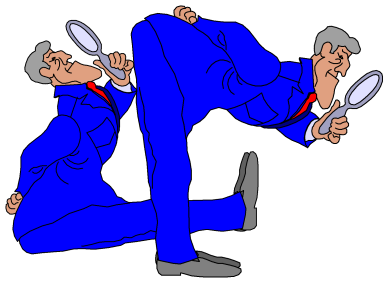
Následující tvrzení uvedeme bez důkazu.

Věta 2.2

Posloupnost vytvořená pomocí lineárního posouvacího registru z nenulového vstupu má maximální periodu právě tehdy, je-li její charakteristický polynom primitivní.

Nalezení primitivních polynomů je netriviální úloha moderní algebry. Poznamenejme pouze, že primitivní polynomy existují pro každé n a že jsou tabelovány.

O čem to bude



Bloky a díry I

Můžeme tedy použít lineární posouvací registry k vytvoření pseudonáhodných posloupností pro kryptografické účely. To je laciné, lineární posouvací registry provádí výpočty velmi rychle – co můžeme ještě víc chtít?

Bloky a díry I

Můžeme tedy použít lineární posouvací registry k vytvoření pseudonáhodných posloupností pro kryptografické účely. To je laciné, lineární posouvací registry provádí výpočty velmi rychle – co můžeme ještě víc chtít?

Nelze popřít, že posloupnosti vytvořené pomocí lineárních posouvacích registrů mají vynikající statistické vlastnosti; a to platí dokonce pro posloupnosti, které vzniknou z relativně krátkých lineárních posouvacích registrů.

Bloky a díry I

Můžeme tedy použít lineární posouvací registry k vytvoření pseudonáhodných posloupností pro kryptografické účely. To je laciné, lineární posouvací registry provádí výpočty velmi rychle – co můžeme ještě víc chtít?

Nelze popřít, že posloupnosti vytvořené pomocí lineárních posouvacích registrů mají vynikající statistické vlastnosti; a to platí dokonce pro posloupnosti, které vzniknou z relativně krátkých lineárních posouvacích registrů.

Ale z kryptologického pohledu mají tyto posloupnosti mimořádně pochybný charakter. To je důsledkem toho, že v případě known-plaintext útoku mu nejsou schopny odolat.

Bloky a díry II

Definujme ***blok délky*** t jako posloupnost tvaru $011\ldots10$ obsahující právě t jedniček. ***Dírou délky*** t je posloupnost tvaru $100\ldots01$ obsahující právě t nul.

Bloky a díry II

Definujme ***blok délky*** t jako posloupnost tvaru $011\ldots10$ obsahující právě t jedniček. ***Dírou délky*** t je posloupnost tvaru $100\ldots01$ obsahující právě t nul.

Platí následující výsledek.

Věta 3.1

Má-li lineární posouvací registr s m registry maximální periodu $2^m - 1$, mají pak výsledné posloupnosti délky $2^m - 1$ následující vlastnosti:

Bloky a díry II

Definujme ***blok délky*** t jako posloupnost tvaru $011\ldots10$ obsahující právě t jedniček. ***Dírou délky*** t je posloupnost tvaru $100\ldots01$ obsahující právě t nul.

Platí následující výsledek.

Věta 3.1

Má-li lineární posouvací registr s m registry maximální periodu $2^m - 1$, mají pak výsledné posloupnosti délky $2^m - 1$ následující vlastnosti:

(a) *obsahuje právě $2^{m-1} - 1$ nul a 2^{m-1} jedniček;*

Bloky a díry II

Definujme ***blok délky*** t jako posloupnost tvaru $011\ldots10$ obsahující právě t jedniček. ***Dírou délky*** t je posloupnost tvaru $100\ldots01$ obsahující právě t nul.

Platí následující výsledek.

Věta 3.1

Má-li lineární posouvací registr s m registry maximální periodu $2^m - 1$, mají pak výsledné posloupnosti délky $2^m - 1$ následující vlastnosti:

(a) *obsahuje právě $2^{m-1} - 1$ nul a 2^{m-1} jedniček;*

(b) *obsahuje pro všechna t taková, že $1 \leq t \leq m - 2$, 2^{m-t-2} bloků délky t a stejný počet děr délky t.*

Bloky a díry III

Důkaz Věty 3.1.

(a): Stav lineárního posouvacího registru lze v každém okamžiku jednoznačně popsat přirozeným číslem z intervalu $[1..2^m - 1]$: stačí vzít příslušnou část výstupní posloupnosti.

Bloky a díry III

Důkaz Věty 3.1.

(a): Stav lineárního posouvacího registru lze v každém okamžiku jednoznačně popsat přirozeným číslem z intervalu $[1..2^m - 1]$: stačí vzít příslušnou část výstupní posloupnosti.

Protože se všechna nenulová čísla z intervalu $[1..2^m - 1]$ musí vyskytnout jako stavy v cyklu maximální délky, výsledek okamžitě dostaneme výsledek (a) spočtením sudých a lichých čísel v této množině.

Bloky a díry III

Důkaz Věty 3.1.

(a): Stav lineárního posouvacího registru lze v každém okamžiku jednoznačně popsat přirozeným číslem z intervalu $[1..2^m - 1]$: stačí vzít příslušnou část výstupní posloupnosti.

Protože se všechna nenulová čísla z intervalu $[1..2^m - 1]$ musí vyskytnout jako stavy v cyklu maximální délky, výsledek okamžitě dostaneme výsledek (a) spočtením sudých a lichých čísel v této množině.

(b): Abychom dokázali (b), poznamenejme, že běh typu **011...10** obsahující právě t jedniček se může vyskytnout jako součást výstupu právě tehdy, když v nějaké části výpočtu je stav lineárního posouvacího registru $011\ldots10x_1x_2\ldots x_{m-t-2}$, kde $x_i \in \{0, 1\}$.

Bloky a díry IV

Důkaz Věty 3.1 - pokračování.

Protože máme právě 2^{m-t-2} stavů tohoto tvaru a protože každý stav je realizován v nějakém okamžiku výpočtu vzhledem k tomu, že lineární posouvací registr má maximální periodu, výsledek (b) pro bloky platí. Zaměníme-li 0 a 1, dostáváme výsledek (b) pro díry. ∎

Known-plaintext útok I

Vraťme se nyní k dešifrování. Je-li

$$\mathbf{M} = M_1 M_2 \ldots$$

zpráva složená z binárních číslic, a je-li

$$\mathbf{Z} = Z_1 Z_2 \ldots$$

posloupnost vyprodukovaná lineárním posouvacím registrem, pak kryptogram **C** je posloupnost

$$\mathbf{C} = C_1 C_2 \ldots,$$

kde

$$C_i = M_i + Z_i \pmod 2 \qquad (1 \leq i < \infty). \quad (3.1)$$

Known-plaintext útok II

Jsou-li tedy M_i a C_i známy, lze Z_i získat triviálně jako

$$Z_i = M_i + X_i \pmod 2 \qquad (1 \leq i < \infty). \quad (3.2)$$

Known-plaintext útok II

Jsou-li tedy M_i a C_i známy, lze Z_i získat triviálně jako

$$Z_i = M_i + X_i \pmod 2 \qquad (1 \leq i < \infty). \quad (3.2)$$

Uvažme nyní lineární posouvací registr s m registry a koeficienty $c_1, c_2, \ldots, c_m$.

Known-plaintext útok II

Jsou-li tedy M_i a C_i známy, lze Z_i získat triviálně jako

$$Z_i = M_i + X_i \pmod 2 \qquad (1 \leq i < \infty). \quad (3.2)$$

Uvažme nyní lineární posouvací registr s m registry a koeficienty $c_1, c_2, \ldots, c_m$.

Jakmile zná nepřítel ***nějakých 2*** m za sebou následujících členů x_i výsledné posloupnosti, je schopen najít tyto koeficienty $c_1, c_2, \ldots, c_m$.

Known-plaintext útok III

Totiž odpovídající systém lineárních rovnic má tvar

$$\begin{bmatrix} Z_m & Z_{m-1} & Z_{m-2} & \cdots & \cdots & Z_2 & Z_1 \\ Z_{m+1} & Z_m & Z_{m-1} & \cdots & \cdots & Z_3 & Z_2 \\ Z_{m+2} & Z_{m+1} & Z_m & \cdots & \cdots & Z_4 & Z_3 \\ \vdots & \vdots & \vdots & \ldots & \ldots & \vdots & \vdots \\ Z_{2m-3} & Z_{2m-4} & Z_{2m-5} & \cdots & \cdots & Z_{m-1} & Z_{m-2} \\ Z_{2m-2} & Z_{2m-3} & Z_{2m-4} & \cdots & \cdots & Z_m & Z_{m-1} \\ Z_{2m-1} & Z_{2m-2} & Z_{2m-3} & \cdots & \cdots & Z_{m+1} & Z_m \end{bmatrix} \cdot \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ \vdots \\ c_{m-2} \\ c_{m-1} \\ c_m \end{bmatrix} = \begin{bmatrix} Z_{m+1} \\ Z_{m+2} \\ Z_{m+3} \\ \vdots \\ Z_{2m-2} \\ Z_{2m-1} \\ Z_{2m} \end{bmatrix}. \quad (3.3)$$

Known-plaintext útok III

Totiž odpovídající systém lineárních rovnic má tvar

$$\begin{bmatrix} Z_m & Z_{m-1} & Z_{m-2} & \cdots & \cdots & Z_2 & Z_1 \\ Z_{m+1} & Z_m & Z_{m-1} & \cdots & \cdots & Z_3 & Z_2 \\ Z_{m+2} & Z_{m+1} & Z_m & \cdots & \cdots & Z_4 & Z_3 \\ \vdots & \vdots & \vdots & \ldots & \ldots & \vdots & \vdots \\ Z_{2m-3} & Z_{2m-4} & Z_{2m-5} & \cdots & \cdots & Z_{m-1} & Z_{m-2} \\ Z_{2m-2} & Z_{2m-3} & Z_{2m-4} & \cdots & \cdots & Z_m & Z_{m-1} \\ Z_{2m-1} & Z_{2m-2} & Z_{2m-3} & \cdots & \cdots & Z_{m+1} & Z_m \end{bmatrix} \cdot \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ \vdots \\ c_{m-2} \\ c_{m-1} \\ c_m \end{bmatrix} = \begin{bmatrix} Z_{m+1} \\ Z_{m+2} \\ Z_{m+3} \\ \vdots \\ Z_{2m-2} \\ Z_{2m-1} \\ Z_{2m} \end{bmatrix}. \quad (3.3)$$

To pak plně určuje šifrovací systém v případě, že matice na levé straně rovnice (3.3) je invertibilní a tudíž platí následující věta.

Known-plaintext útok IV

Věta 3.2

Je-li posloupnost bitů $\mathbf{Z} = Z_1 Z_2 \ldots$ *generována regulárním lineárním posouvacím registrem R délky m a neexistuje-li kratší lineární posouvací registr generující tuto posloupnost, pak je charakteristický polynom lineárního posouvacího registru R jednoznačně určen 2m za sebou jdoucími členy této posloupnosti.*

Known-plaintext útok IV

Věta 3.2

Je-li posloupnost bitů $\mathbf{Z} = Z_1 Z_2 \ldots$ *generována regulárním lineárním posouvacím registrem R délky m a neexistuje-li kratší lineární posouvací registr generující tuto posloupnost, pak je charakteristický polynom lineárního posouvacího registru R jednoznačně určen 2m za sebou jdoucími členy této posloupnosti.*

Důkaz. Stačí ověřit že matice $\mathbf{A}$ na levé straně rovnice (3.3) je invertibilní. Předpokládejme opak. Pak nutně její sloupce jsou lineárně závislé. Přitom sloupce nejsou nic jiného než stavy systému: $\mathbf{X}(0), \mathbf{X}(1), \ldots, \mathbf{X}(m-1)$, máme tedy lineární kombinaci

$$\sum_{i=0}^{m-1} b_i \mathbf{X}(i) = \mathbf{0}. \tag{3.4}$$

Known-plaintext útok V

Přitom koeficienty $b_i \in \{0, 1\}$ nejsou všechny nulové.

Known-plaintext útok V

Přitom koeficienty $b_i \in \{0, 1\}$ nejsou všechny nulové.
Položme

$$k = max\{i : b_i \neq 0\}.$$

Pak $k \leq m - 1$ a nutně (protože pracujeme mod 2) máme

$$\sum_{i=0}^{k-1} b_i \mathbf{X}(i) = \mathbf{X}(k). \tag{3.5}$$

Known-plaintext útok V

Přitom koeficienty $b_i \in \{0, 1\}$ nejsou všechny nulové. Položme

$$k = max\{i : b_i \neq 0\}.$$

Pak $k \leq m - 1$ a nutně (protože pracujeme mod 2) máme

$$\sum_{i=0}^{k-1} b_i \mathbf{X}(i) = \mathbf{X}(k). \tag{3.5}$$

Buď nyní $\mathbf{C}$ matice lineárního posouvacího registru R. Pak pro každé $t \geq 0$ platí

$$\mathbf{X}(t+k) = \mathbf{C}^t \mathbf{X}(k) = \sum_{i=0}^{k-1} b_i \mathbf{C}^t \mathbf{X}(i) = \sum_{i=0}^{k-1} b_i \mathbf{X}(i+t). \tag{3.6}$$

Known-plaintext útok VI

Speciálně tedy pro $t \geq 1$ platí

$$Z_{t+k} = \sum_{i=0}^{k-1} b_i Z_{t+i}. \tag{3.7}$$

Known-plaintext útok VI

Speciálně tedy pro $t \geq 1$ platí

$$Z_{t+k} = \sum_{i=0}^{k-1} b_i Z_{t+i}. \tag{3.7}$$

Tudíž posloupnost $\mathbf{Z} = Z_1 Z_2 \ldots$ je generována lineárním posouvacím registrem R' délky k (přičemž příslušný charakteristický polynom lineárního posouvacího registru R' má koeficienty $b_0, \ldots, b_{k-1}$).

Known-plaintext útok VI

Speciálně tedy pro $t \geq 1$ platí

$$Z_{t+k} = \sum_{i=0}^{k-1} b_i Z_{t+i}. \tag{3.7}$$

Tudíž posloupnost $\mathbf{Z} = Z_1 Z_2 \ldots$ je generována lineárním posouvacím registrem R' délky k (přičemž příslušný charakteristický polynom lineárního posouvacího registru R' má koeficienty $b_0, \ldots, b_{k-1}$).

To je ale spor s minimalitou m.

Known-plaintext útok VI

Speciálně tedy pro $t \geq 1$ platí

$$Z_{t+k} = \sum_{i=0}^{k-1} b_i Z_{t+i}. \tag{3.7}$$

Tudíž posloupnost $\mathbf{Z} = Z_1 Z_2 \ldots$ je generována lineárním posouvacím registrem R' délky k (přičemž příslušný charakteristický polynom lineárního posouvacího registru R' má koeficienty $b_0, \ldots, b_{k-1}$).

To je ale spor s minimalitou m.

Poznamenejme, že výše uvedené je překvapující výsledek. Totiž to znamená, že můžeme posloupnost více než milionu bitů (přesněji $1048575 = 2^{20} - 1$ bitů) rekonstruovat ze znalosti pouhých 40 výstupních bitů.

Known-plaintext útok VII

Důsledek 3.3

Užití posloupností vytvořených pomocí lineárního posouvacího registru není bezpečné proti known-plaintext útoku.

Known-plaintext útok VII

Důsledek 3.3

Užití posloupností vytvořených pomocí lineárního posouvacího registru není bezpečné proti known-plaintext útoku.

Důkaz. Předpokládejme, že odesilatel Alice a příjemce Bob používají proudovou šifru, jejímž klíčem je výstup z lineárního posouvacího registru R délky m. Nechť útočník Eve zná část zdrojového textu o délce $2m$, řekněme $M_{i+1}, M_{i+2}, \ldots, M_{i+2m}$. Pokud Eve zachytí odpovídající část šifrového textu $C_{i+1}, C_{i+2}, \ldots, C_{i+2m}$, pak samozřejmě zná i odpovídající část klíče $Z_{i+1}, Z_{i+2}, \ldots, Z_{i+2m}$.

Known-plaintext útok VII

Důsledek 3.3

Užití posloupností vytvořených pomocí lineárního posouvacího registru není bezpečné proti known-plaintext útoku.

Důkaz. Předpokládejme, že odesilatel Alice a příjemce Bob používají proudovou šifru, jejímž klíčem je výstup z lineárního posouvacího registru R délky m. Nechť útočník Eve zná část zdrojového textu o délce $2m$, řekněme $M_{i+1}, M_{i+2}, \ldots, M_{i+2m}$. Pokud Eve zachytí odpovídající část šifrového textu $C_{i+1}, C_{i+2}, \ldots, C_{i+2m}$, pak samozřejmě zná i odpovídající část klíče $Z_{i+1}, Z_{i+2}, \ldots, Z_{i+2m}$. Dle předchozí věty je tedy Eve schopna určit koeficienty charakteristického polynomu lineárního posouvacího registru R, tj. zkonstruovat R. Může tedy, vezme-li za počáteční základ $Z_{i+1}, Z_{i+2}, \ldots, Z_{i+2m}$ vygenerovat všechny předchozí i následující členy klíče. Eve je tedy schopna dešifrovat zbytek zprávy.

Known-plaintext útok VIII

Chceme-li zachovat hezké vlastnosti lineárních registrů a zároveň zajistit větší stupeň bezpečnosti, použijeme v rovnici 2.2 nelineární funkci. Skutečně je tomu tak, že většina dnes používaných algoritmů je založena na nelineárních posouvacích registrech, ačkoliv bychom neměli zapomenout na DES případně Triple-DES.

Known-plaintext útok VIII

Chceme-li zachovat hezké vlastnosti lineárních registrů a zároveň zajistit větší stupeň bezpečnosti, použijeme v rovnici 2.2 nelineární funkci. Skutečně je tomu tak, že většina dnes používaných algoritmů je založena na nelineárních posouvacích registrech, ačkoliv bychom neměli zapomenout na DES případně Triple-DES.

Zvláště rafinovaná metoda je tzv. ***shrinking generator***. V tomto případě použijeme dva lineární posouvací registry, které pracují ve stejném taktu. Budeme se řídit předpisem, že použijeme právě ty výstupní bity druhého lineárního posouvacího registru, pro které je zároveň příslušná hodnota prvního lineárního posouvacího registru rovná 1.

Položme si následující otázku: ***Lze rozumně měřit kryptologickou kvalitu posloupnosti nul a jedniček?*** Viděli jsme, že se k tomuto perioda sotva hodí.

Known-plaintext útok IX

Proto se používá pojem ***lineární složitosti dané posloupnosti***, jakožto nejkratší délka takového lineárního posouvacího registru, že daná posloupnost je vytvořena jakožto část výstupu tohoto lineárního posouvacího registru. Čím větší je lineární složitosti dané posloupnosti, tím lépe se tato posloupnost hodí pro kryptografické účely. Například metoda shrinking generator nám garantuje vysokou lineární složitost.

Known-plaintext útok IX

Proto se používá pojem ***lineární složitosti dané posloupnosti***, jakožto nejkratší délka takového lineárního posouvacího registru, že daná posloupnost je vytvořena jakožto část výstupu tohoto lineárního posouvacího registru. Čím větší je lineární složitosti dané posloupnosti, tím lépe se tato posloupnost hodí pro kryptografické účely. Například metoda shrinking generator nám garantuje vysokou lineární složitost.

Udělejme si představu o náročnosti hádání klíče. Pravděpodobnost uhodnutí 64-bitového klíče je $1/2^{64}$, což je samozřejmě kladné číslo. Porovnejme si toto číslo s jinými velikostmi.

Known-plaintext útok X

Připomeňme, že

$$2^{64} \approx 1,84 \cdot 10^{19}.$$

Je tedy uhodnutí 64-bitového klíče stejně hodnotné, jakožto vybrání předem určeného prvku z množiny o 10 trilionech prvků, což je více než všech možných partnerských párů na zeměkouli (v současnosti nás je asi $6 \cdot 10^9$).

Known-plaintext útok X

Připomeňme, že

$$2^{64} \approx 1,84 \cdot 10^{19}.$$

Je tedy uhodnutí 64-bitového klíče stejně hodnotné, jakožto vybrání předem určeného prvku z množiny o 10 trilionech prvků, což je více než všech možných partnerských párů na zeměkouli (v současnosti nás je asi $6 \cdot 10^9$). Podobně počet všech 256-bitových klíčů lze odhadout jako

$$2^{256} \approx 1,15 \cdot 10^{77},$$

což je číslo větší než počet elementárních částic v našem vesmíru. Je tedy jasné, že útok hrubou silou nám bude k ničemu.