O rozkládání a skládání tajemství
About splitting and reconstruction of secret

CZ:
Motivace

Rozkládání a skládání tajemství je systém, pomocí kterého je možné informaci (číslo, dále jen „tajemství“ S) „rozložit“ („split“) na více částí, přičemž pouze nějaká podmnožina (možno i všechny části na které bylo tajemství rozloženo, záleží na konfiguraci systému - „threshold“ T - česky se přidržím označení podmnožina) umožňuje „složit“ („reconstruct“) tyto části zpět a získat tak původní tajemství.

Lze si představit, že má odesílatel využívající tento systém A různých připojení k síti, přičemž ví, že B jich je odposlouchávaných (některá tedy nejsou, odesílatel neví která, v praxi se stanoví číslo B podle pravděpodobnosti z čísla A). Odesílatel rozloží tajemství na A částí s tím, že pouze podmnožina (B+1) částí složí zpět původní tajemství. Takto rozložené části pošle těmito připojeními adresátovi a ten složí zpět tajemství. B a méně částí neumožňuje získat tajemství žádným způsobem. V implementaci na https://github.com/cenekSvoboda/SSRI bylo zvoleno A=5 a B=2.

Matematika

Systém využívá Lagrangeovu interpolaci. Tajemství je y-ová souřadnice bodu na křivce dané polynomem s x-ovou souřadnicí x=0, což zjednodušuje interpolaci.

Systém rozkládání

Kromě vstupu do systému (tajemství) byly do implementace rozkládání přidány dvě numerické konstanty a jedna boolean konstanta:
coefRange1 – konstanta určující maximální x-ovou vzdálenost bodu na křivce od x=0
coefRange2 – konstanta určující maximální koeficient u proměnné x určitého řádu polynomu
enableHighDistance – je-li tato konstanta nastavena na pravdivou hodnotu, systém do polynomu vkládá x-ové souřadnice vzdálené od x=0 minimálně o coefRange1/2.

Polynom je tedy v následujícím tvaru:

$y = S + a_1x + a_2x^2 + \ldots + a_Ax^A$

Systém v prvním cyklu náhodně stanoví všem částem všechny x-ové souřadnice (podle coefRange1 a enableHighDistance).
Ve druhém cyklu stanoví náhodně koeficienty a_1 až a_A (podle coefRange2).
Ve třetím cyklu vypočítá y-ové hodnoty částí podle výše uvedeného polynomu.

Výstupem je dvojrozměrné pole ve tvaru $[[x_0,y_0],[x_1,y_1],\ldots,[x_{A-1},y_{A-1}]]$ (použita JSON notace).

Zjednodušení interpolace

Hledáme y=S, kde x=0.

$$y = y_0 \times \frac{0-x_1}{x_0-x_1} \times \frac{0-x_2}{x_0-x_2} \times \ldots + y_1 \times \frac{0-x_0}{x_1-x_0} \times \frac{0-x_2}{x_1-x_2} \times \ldots + \ldots$$

Systém skládání

Vstupem do systému je dvojrozměrné pole ze systému rozkládání, resp. jeho část $[[x_0,y_0],[x_1,y_1],\ldots,[x_B,y_B]]$, přičemž nezáleží na pořadí.

V implementaci jsou dvakrát dva vnořené cykly.
V prvním vnořeném cyklu je pomocné pole lagrangeFreeCoef naplněno součinem x-ových souřadnic s tím, že vnitřní cyklus vynechává násobení při shodném indexu s vnějším cyklem. Násobení není realizováno se zápornou hodnotou, na systém to nemá vliv; zda je výsledek opačným číslem se určí až na závěr systému skládání.
Ve druhém vnořeném cyklu jsou prvky pomocného pole děleny rozdíly dle vzorce z kapitoly „Zjednodušení interpolace“ s obdobným vynecháváním jako v prvním vnořeném cyklu.

Ve třetím cyklu je pomocné pole lagrangeFreeCoef násobeno y-ovými souřadnicemi a sečteno. Tím získáme výsledné tajemství.

Implementace

Systém je naimplementován v programovacím jazyce JavaScript (ECMAScript), takže proměnné jsou typu „double precision floating point numbers, following the international IEEE 754 standard “ (64bit). Algoritmy provádějí násobení a dělení, během kterého může dojít k významné ztrátě přesnosti proměnných (proměnná začne být v exponenciálním tvaru), takže byl navržen útok hrubou silou na celý systém během rozkládání. Systém se pokusí náhodně vygenerovat všechny části a následně se je pokusí složit. Pokud dojde při skládání (které je neoddělitelnou součástí rozkládání) v některé proměnné ke změně na jinou, než 52bitovou reprezentaci čísla (číslo významně ztratí přesnost), systém nevrátí původní tajemství. Systém tedy navrhne nové části, které se pokusí znovu složit. To se opakuje dokud všechny permutace všech kombinací částí nevracejí původní tajemství. Systém tedy nevyžaduje, aby byly části v nějakém pořadí. Systém umožňuje použití záporného čísla jako tajemství a také může záporné souřadnice vygenerovat.

Během implementace bylo experimentálně zjištěno, že systém může vracet tajemství s malou nepřesností. Ošetřeno zaokrouhlováním.

Právo

Použitá interpolace byla vynalezena před více jak 70-ti lety, takže matematické formule patří mezi díla v „public domain“. Celý implementovaný systém je tak rovněž darován do „public domain“. Tento dokument do „public domain“ není darován.

EN:
Motivation

Secret splitting and reconstruction is a system, that allows the user to „split“ information (number, secret S) into more pieces, where only threshold T pieces allows to „reconstruct“ those pieces back and get the original secret.

It's possible to imagine, that sender using this system has A different connections to network, where she/he knows, that B of them are spied (some of them are not, B could be determined using probability from A). Sender splits the secret to A pieces. Only B+1 pieces allow receiver to reconstruct the original secret. A=5 and B=2 were chosen in implementation on https://github.com/cenekSvoboda/SSRI.

Mathematics

System uses Lagrange's interpolation. Secret is the y coordinate of point on curve given by polynomial with x coordinate x=0. This simplifies interpolation.

Splitting system

Aside from the input to the system (secret), two numeric constants and one boolean constant were added to implementation:

coefRange1 – constant determining maximal x distance of point on curve from x=0
coefRange2 – constant determining maximal coeficient multiplying variable x of some order in polynomial
enableHighDistance – if this constant is set to true, system puts into polynomial x coordinates distant from x=0 at least to coefRange1/2

Polynomial is in this form:

$y = S + a_1x + a_2x^2 + \ldots + a_Ax^A$

System in first cycle sets random x coordinates to all pieces (based on coefRange1 and enableHighDistance).
In second cycle it sets random coeficients a_1 to a_A (based on coefRange2).
In third cycle it calculates y coordinates of pieces based on the polynomial above.

The output is two dimensional array in form $[[x_0,y_0],[x_1,y_1],\ldots,[x_{A-1},y_{A-1}]]$ (used JSON notation).

Simplifying interpolation

We seek y=S, where x=0.

$$y=y_0\times\frac{0-x_1}{x_0-x_1}\times\frac{0-x_2}{x_0-x_2}\times\ldots+y_1\times\frac{0-x_0}{x_1-x_0}\times\frac{0-x_2}{x_1-x_2}\times\ldots+\ldots$$

Reconstruction system

Input to the system is two dimensional array from splitting system, or its part $[[x_0,y_0],[x_1,y_1],\ldots,[x_B,y_B]]$, order doesn't matter.

In implementation, there are two nested cycles.

In first nested cycle the support array lagrangeFreeCoef is filled with multiplication of x cooridnates. Inner cycle doesn't multiply on equal index with outer cycle. Multiplication is not realised with negative value, doesn't matter for the system; if the output is additive inverse is determined at the end of reconstruction system.
In second nested cycle the support array lagrangeFreeCoef elements are divided by differences from the chapter „Simplyfying interpolation“, the division doesn't happen on equal index of inner and outer cycle.

In third cycle the support array lagrangeFreeCoef is multiplied by y coordinates and added together. We get the original secret.

Implementation

System is implemented in programming language JavaScript (ECMAScript), so the variables are “double precision floating point numbers, following the international IEEE 754 standard” (64bit). Algorithms do multiplication and division, during which the precision may be lost significantly (the variable gets into exponential form). That's the reason why bruteforce attack was implemented on the whole system during splitting. System tries to generate all pieces and it tries to reconstruct them. If in any element anywhere in any array the element gets into other than 52bit representation of the number (the number significantly loses precision), reconstruction system doesn't return original secret. System makes new pieces and tries to reconstruct them again. This happens as long as all permutations of all combinations of pieces don't return the original secret. System doesn't require pieces to be in any order. Negative numbers are allowed in all parts of the system.

During implementation it was experimentally verified, that the system may return the original secret with little loss of precision. This was taken in consideration and rounding was implemented.

Law

Used interpolation was created more than 70 years ago, so the formulas belong to public domain. Therefore the whole implemented system (code) was given to public domain. This document is not in public domain.

References:

https://en.wikipedia.org/wiki/Lagrange_polynomial

Archer, Branden and Weisstein, Eric W. "Lagrange Interpolating Polynomial." From *MathWorld*--A Wolfram Web Resource. http://mathworld.wolfram.com/LagrangeInterpolatingPolynomial.html

https://www.w3schools.com/js/js_numbers.asp

https://en.wikipedia.org/wiki/Shamir%27s_Secret_Sharing

Document version: 2.1.1