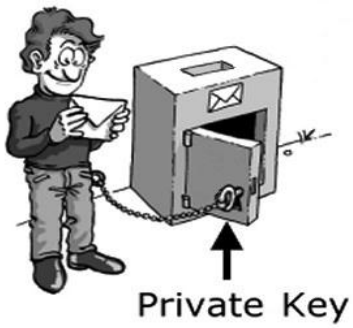
អត្ថបទដកស្រង់ចេញពី

Klein Project Blog

កូនសោសម្ងាត់សាធារណៈ *Cryptography*

(Public-key Cryptography)



រៀបរៀងដោយអ្នកនិពន្ធ: Graeme L. Cohen, Steven Galbraith និង *EdoardoPersichetti* ។

តើយើងអាចផ្ញើកាតឥណទាន *(Credit Card)* របស់យើងដោយសុខសុវត្ថិភាពលើប្រព័ន្ធ *Internet* ឬការប្រើទូរស័ព្ទទំនើបពេលអ្នកដ៏ទៃអាចស្តាប់យកសាររបស់យើងដោយរបៀបណា? តើយើងអាចទុកចិត្តកម្មវិធីដែល *Updates* ពេលយើងដឹងថាកុំព្យូទ័រមានវីរុស *(Viruses)* គឺជារឿងធម្មតាដោយរបៀបណា?

Cryptography (គឺសិក្សាអំពីម៉ាស៊ីនសម្រាប់ការទាក់ទងដ៏រឹងមាំក្នុងវត្តមាននៃសត្រូវ) គឺផ្តល់ចម្លើយនៃសំនួរ ទាំងនេះ និងមុខវិជ្ជាគណិតវិទ្យាសម្រាប់ផ្តល់តួនាទីរបស់វា។

ប្រវត្តិខ្លី

ការទាក់ទងដ៏មាំមួនបានមានសារៈសំខាន់សម្រាប់មួយពាន់ឆ្នាំដែលទាំងនេះជាភស្តុតាងរបស់ *Julius Caesar* បានប្រើវិធីក្រាបសម្ងាត់ដ៏សាមញ្ញសម្រាប់ទំនាក់ទំនងជាមួយឧត្តមសេនីរបស់គាត់។ គោងការណ៍ គឺបានដឹងដូចជា *Caesar* សរសេរសម្ងាត់ *(Caesarcipher)* ក៏ដោយហើយវាកើតឡើងសម្រាប់ផ្លាស់ប្តូរអក្សរ ក្នុងសារ *(message)* ដោយចំនួនត្រឹមត្រូវនៃទីតាំងពេលបានឯកសារថ្មីដែលគេឲ្យហៅថា អត្ថបទសរសេរសម្ងាត់ *(ciphertext)*។ សារអាចបានមកវិញនៅពេលអ្នកដ៏ទៃបានបញ្ចប់វាហើយដោយការត្រឡប់វិធីលេខទាំងនេះ។

គំនិតសំខាន់គឺថាទាំងអ្នកផ្ញើនិងទាំងអ្នកទទួលត្រូវដឹងតម្លៃសម្ងាត់មួយ (ក្នុងករណីនេះ គឺចំនួននៃទីតាំង) គឺថាជាការស្មានដែលមនុស្សដ៏ទៃមិនដឹងទោះបីព្យាយាមសិក្សាពីសារក៏ដោយ។ ចំនែកឯតម្លៃសម្ងាត់នោះហៅថា កូនសោ *(Key)*។ នៅក្នុងករណីកូនសោ *Caesar cipher* ខាងលើជាចំនួនរវាងពី 0 ទៅ 26 ។ ការសរសេរលេខសម្ងាត់ *algorithm Enc* ដាក់បញ្ចូលក្នុងសារ *M* និងកូនសោ K ជាឧទាហរណ៍ $Enc_3(Hello) = Khoor$។ នៅ *algorithm Dec* ដាក់បញ្ចូលអត្ថបទសរសេរ *(cipher text) C* និងកូនសោដូចគ្នា *K* ជាឧទាហរណ៍ $Dec_3(Khoor) = Hello$។

ប្រព័ន្ធលេខសម្ងាត់ *(Crytosystems)* ដែលមានកូនសោដូចគ្នា គឺត្រូវបានគេប្រើសម្រាប់ការសរសេរ លេខសម្ងាត់ *(encrypting)* និងបកស្រាយលេខសម្ងាត់ *(decrypting)* មានឈ្មោះថាគោលការណ៍កូនសោតែមួយ *(Symmetric-Key schemes)*។

គោលការណ៍ *Caesar* ខ្លួនឯង គឺសាមញ្ញបំផុតដែលធ្វើឲ្យផុតភ័យក្នុងពិភពលោកទំនើបនេះ ប៉ុន្តែក្នុងសម័យទំនើបមានអ្នកសរសេរកូនសោសម្ងាត់តែមួយ *(symmetric-key cipher)* គឺថាបច្ចុប្បន្នបានប្រើក្នុងស្ថាប័នជាច្រើន ជាឧទាហរណ៍ដ៏ល្បីល្បាញរបស់ *AES* គឺមួយក្នុងចំណោមក្រុមអ្នកគ្រប់គ្រងស្តង់ដាររបស់ សហរដ្ឋអាមេ

វិចសម្រាប់ទិន្នន័យផ្ទេរ។ ប្រព័ន្ធទាំងនេះ គឺមានសមត្ថភាពមាំមួន ប៉ុន្តែមានបញ្ហាមួយទាំងអ្នកផ្ញើ និងអ្នកទទួលត្រូវបែងចែកជាសម្ងាត់រួចហើយ។ តើយើងអាចទំនាក់ទំងនជាសម្ងាត់លើប្រព័ន្ធ *Internet* ជាមួយមុនស្សដែលយើងមិនដែលបានជួបដោយរបៀបណា?

គំនិតធ្វើឲ្យឆ្ងល់មួយហាត្រូវបានគេហៅថា *Public-Key cryptography* ដែលបានបញ្ចូលជា សមាជិកថ្មីឆ្នាំ 1976 ក្នុងក្រដាស *"New directions in cryptography"* ដោយលោក *Whitfield Diffie* និង *Martin Hellman* បានបកស្រាយពីបញ្ហានេះ។ នៅក្នុងការដាក់ជំនួសនៃការប្រើកូនសោដូចគ្នាសម្រាប់ការសរសេរ លេខសម្ងាត់ *(encrypting)* និងការបកស្រាយលេខសម្ងាត់ *(decrypting)* ទាំងនេះជាកូនសៅសាធារណៈ ដែលអាចរកបានដល់អ្នកប្រើប្រាស់មិនទាន់បានឃើញជាក់ស្តែងនិងកូនសៅឯកជនដែលសម្ងាត់គង់នៅដល់អ្នកប្រើប្រាស់មិនទូទៅ។ ម៉្យាងទៀតអ្នកដ៏ទៃអាចធ្វើសារប៉ុន្តែមានមនុស្សតែម្នាក់គត់អាចទទួលវា ពោលគឺអ្នកណាខ្លះ អាចសរសេរនៅក្នុងតាមកន្លែងចន្លោះតូចៗប៉ុន្តែមានមុនស្សម្នាក់គត់ដែលមានកូនសៅសម្រាប់ប្រអប់សំបុត្រអាចទទួលបានសំបុត្រ។ នៅក្នុងការទាក់ទងសម្ងាត់ជាមួយ *Alice* មួយអាចរកមើលកូនសោជាសាធារណៈរបស់ ពួកគេ និងប្រើវាបង្កើតអត្ថបទសម្ងាត់ *(cipher text)* មួយ។ មានតែ *Alice* ម្នាក់គត់ ដែលអាចបកស្រាយ លេខសម្ងាត់ក្នុងអត្ថបទសម្ងាត់ ដូចជាគាត់បានដឹងកូនសៅអញ្ចឹង។

ប្រព័ន្ធសម្ងាត់កូនសោសាធារណៈ ត្រូវតែជាមូលដ្ឋានគ្រឹះនៅក្នុងបញ្ហាដែលគិតលេខគឺថាដោះស្រាយ ពិបាក។ គណិតវិទ្យា ជាអ្នកដោះស្រាយបញ្ហានេះ សម្រាប់ជាឧទាហរណ៍ ប្រព័ន្ធសម្ងាត់ *RSA* ជាមូលដ្ឋានគ្រឹះ ដ៏លំបាកសម្រាប់ក្នុងការស្វែងរកកត្តាបឋមនៃចំនួនគត់ដែលវែង។

ទ្រឹស្តីចំនួនមួយចំនួន

មុនរៀបរាប់ពីកូនសោសាធារណៈ *RSA* ដ៏ល្បីល្បាញណាស់ក្នុងការផែនការណ៍បកស្រាយលេខសម្ងាត់ *(encryption scheme)* យើងចាំបាច់ត្រូវការអភិវឌ្ឍលទ្ធផលនៅក្នុងទ្រឹស្តីចំនួនសិន។ នេះត្រូវការភាពស្គាល់ច្បាស់សមហេតុសមផលជាមួយទ្រឹស្តីទ្វេធា *(binomial theorem)* និងប្រមាណវិធីនព្វន្ត *modular* ។

នៅក្នុងប្រមាណវិធីនព្វន្ត *modular* យើងប្រមែប្រមូលថ្នាក់ដែលមានសំណល់ដូចគ្នា បន្ទាប់ពី ការចែកដោយចំនួនពិតប្រាដក P មួយដែលគេត្រូវបានហៅថា *modular* ។ ដូចឧទាហរណ៍ បើ $P = 7$ យើងបាន 9 គឺស្ថិតនៅថ្នាក់ 2 ហើយ 12 និង 19 មានថ្នាក់ដូចគ្នាពោលគឺថ្នាក់ 5។ យើងអាចសរសេរថា $9 \equiv 2(mod7)$ និង $12 \equiv 19 \equiv 5(mod7)$។ វាគឺងាយក្នុងការដែលឃើញថាមានតែលេខ 7 គត់មានថ្នាក់អាចកើតឡើង ពោលគឺពេលធ្វើការ *modulo 7* នោះមានថ្នាក់រៀងគ្នាគឺ $\{0,1,2,3,4,5,6\}$ វាជាលទ្ធផលដែលកើតពី ធ្វើផលបូក និងផលគុណ *modulo 7* ដោយការបង្ហូមយ៉ាងសាមញ្ញនូវលទ្ធផលបន្ទាប់ពីការធ្វើប្រមាណវិធីធម្មតា គឺថា $3 + 8 = 11 \equiv 4(mod7)$ និង $3 \cdot 5 = 15 \equiv 1(mod7)$ ។

ចំណាំ៖ បើ $a \equiv b\ (\ mod\ p\ ) \Rightarrow (a - b)$ជាផលគុណនៃ p ។

បើតាង p និង q ជាចំនួនបឋមដែល $p \neq q$ និង $t > 0$ ជាចំនួនគត់ដែលមិនស្មើផលគុណនៃ p ឬ q ឡើញនោះយើងនឹងបង្ហាញរូបមន្ត $t^{(p-1)(q-1)} \equiv 1(mod\ pq)$ ជាការពិត។ នេះជារូបមន្តពិសេសនៃទ្រឹស្តី *Fermat-Euler* ។ ការបកស្រាយចាប់ផ្តើមជាមួយរូបមន្តមេគុណទេធា $\binom{p}{i} = \frac{p!}{i!(p-i)!}$ ដែល i ជាចំនួនគត់ ជាមួយ$0 < i < p$។ តាម

ទម្រង់នេះអាចសរសេរបាន $i!\binom{p}{i} = \frac{p!}{(p-i)!} = p(p-1)(p-2)\cdots(p-i+1)$ គឺវាត្រូវតែ p ជាតួចែក $\binom{p}{i}$ ដោយ p ចែកដាច់ផ្នែកខាងស្តាំ ប៉ុន្តែមិនចែកដាច់នឹង $i!$ ទេ។ នោះសម្រាប់ចំនួនគត់ A និង B តាមទ្រឹស្តីបទទ្វេធា ៖

$$(A+B)^p = A^p + \binom{p}{1}A^{p-1}B + \cdots + B^P \equiv A^p + B^P (mod\ p)$$

ការយកចំនួនគត់ C មកបន្ថែមដោយជាបីតួគឺ៖

$$(A+B+C)^p \equiv (A+B)^p + C^P \equiv A^p + B^p + C^p (mod\ p)$$

នៅក្នុងវិធីនេះដែលអនុវត្តន៍ទៅដល់ t តួគឺ $A_1, A_2, \cdots, A_t$ នោះ

$$(A_1 + A_2 + \cdots + A_t)^p \equiv A_1^p + A_2^p + \cdots + A_t^p\ (mod\ p)$$

ឥឡូវយក$A_1 = A_2 = \cdots = A_t = 1$នោះយើងបាន $t^p = t\ (mod\ p)$ នេះគឺជាសមីការដែលមានន័យថា $t^p - t = t(t^{p-1} - t)$ ជាផលគុណមួយនៃ p តែ p មិនចែកដាច់នឹង t ទេហើយ p ជាចំនួនបឋម ដូចនេះវាដូចខាងក្រោម៖

$$t^{p-1} = 1\ (mod\ p)$$

តាមលក្ខណៈនៃ *modulo* យើងលើកអង្គទាំងពីរជាស្វ័យគុណ q - *1* :

$$t^{(p-1)(q-1)} = 1\ (mod\ p)$$

ដោយធ្វើនូវសេចក្តីសំអាងដដែលឡើងវិញ ប៉ុន្តែប្រើ q ជំនួស p

យើងបាន៖ $t^{(p-1)(q-1)} = 1\ (mod\ q)$ ទាំងនោះអាចនិយាយបានថាដែលចំនួនគត់ h និង k រៀងគ្នា នូវលទ្ធផលខាងក្រោម៖

$$t^{(p-1)(q-1)} = 1 + ph\,, \qquad t^{(p-1)(q-1)} = 1 + qk$$

ដូចនេះ $ph = kq$ នោះ p ចែកដាច់នឹង k (ដោយ p *និង* q ជាចំនួនបឋមផ្សេងគ្នា) នោះនិយាយបានថា $k = pl$ ដែល l ជាចំនួនគត់មួយចំនួនបើដូច្នោះ គេបាន $t^{(p-1)(q-1)} = 1 + pql$ ឬ $t^{(p-1)(q-1)} \equiv 1\ (mod\ pq)$

ទាំងនេះជាសម្រាយសម្រាប់រូបមន្តដែលត្រូវបានបំពេញ។

ប្រព័ន្ធសម្ងាត់ *RSA*

ពាក្យកាត់នៃ *RSA* មកពីពាក្យពេញថា *Rivest, Shamir* និង *Adleman* ដែលនេះជាសំណើដំបូងសម្រាប់គ្រោងការណ៍នៅក្នុងឆ្នាំ 1977 ដែលវាមានដូចខាងក្រោម៖

ជាដំបូងជ្រើសរើសចំនួនបឋមពីរផ្សេងគ្នាគឺ p និង q បន្ទាប់មកចាប់ផ្តើមគណនា $N = pq$ & $\phi = (p-1)(q-1)$ ។ រើសយកចំនួនគត់ e មួយដោយបិទភ្នែករវាងលេខ 3 ទៅ N – 2 ប៉ុន្តែដូច្នោះថាចំនួន e & ϕ មិនមានកត្តាបឋមដែលរួមគ្នាទេ។ កូនសោសាធារណៈជាផ្នែកមួយនៃ (N,e) ចំនែកឯ កូនសៅឯក

ជន d ដែលអ្នកផ្ញើនឹងរក្សាទុកដោយសម្ងាត់គឺជាការគណនា $d \equiv e^{-1} (\mathrm{mod}\, \phi)$ ដែលជាចំនួនដូចជា $de \equiv 1 (\mathrm{mod}\, \phi)$ (នេះវាងាយស្រួលដោយប្រើ *algorithm* របស់ *Euclid*)។

សម្រាប់ជាឧទាហរណ៍យក៖ $N = 437 = 19 \cdot 23,\ \phi = 396, e = 5\ \&\ d = 317$ ។ សារនៅក្នុងប្រព័ន្ធ *RSA* ជាចំនួនគត់ x ដែលថា $0 \prec x \prec N$ វាអាចនឹងមិនជាខាងដើមក្នុងការធ្វើឲ្យច្បាស់នូវរបៀបសរសេរសារអត្ថបទសម្ងាត់ជាមួយគ្រោងការណ៍ដែលប្រើចំនួនគត់។ទោះបីជាយ៉ាងណាក៏ដោយគោងការណ៍ក្រាបសម្ងាត់ *(cryptographic)* ប្រដាប់ប្រដានៅក្នុងកុំព្យូទ័រនិងឯកសារទាំងអស់គឺត្រូវជាទិន្នន័យប្រព័ន្ធគោល 2 *(binary)* ដែលអាចប្រើ សរសេរចំនួនគត់ជាអក្សរសម្ងាត់បាន។

របៀបសរសេរលេខសម្ងាត់គឺដូចខាងក្រោម៖

ឧបមាថា *Bob* ចង់ផ្ញើសារ x ទៅ *Alice* ។ គាត់រកមើលកូនសៅសាធារណៈ (N, e) របស់ *Alice* គឺគាត់គណនា $y = x^e (\mathrm{mod}\, N)$ និងផ្ញើ y នេះទៅ *Alice*។ អត្ថបទសរសេរបកប្រែលេខសម្ងាត់របស់ y គឺ *Alice* យកប្រើ នៃកូនសោឯកជន d គ្រាន់តែគណនា $x = y^d (\mathrm{mod}\, N)$ ។

វាជាចំនុចសំខាន់ទៅចំនុចក្រៅថាពេលដែល $e, d\ \&\ N$ ជាចំនួនធំណាស់គឺវាអាចមានសមត្ថភាពគណនាសម្រាប់ឧទាហរណ៍ $y^d (\mathrm{mod}\, N)$ ដែលក្នុងការប្រើម៉ាស៊ីនត្រូវបានហៅថាស្វ័យគុណ *modulo (modular exponentiation)*។

ការបកប្រែលេខសម្ងាត់ជាការការងារចាប់ផ្តើមដោយ $de = 1 + k\phi$ គ្រប់ចំនួនគត់ k ហើយដូចជាយើងបានបង្ហាញអំពីរូបមន្ត៖ $x^{\phi} = x^{(p-1)(q-1)} \equiv 1 (\mathrm{mod}\, N)$ (ក្នុងករណីប្រតិបត្តិពេល x ជាពហុគុណនៃ p និង q មិនអាចបានទេ) យើងបាន $y^d \equiv (x^e)^d = x^{1+\phi k} = x(x^{\phi})^k \equiv x \cdot 1^k = x (\mathrm{mod}\, N)$ ។

អ្នកប្រើប្រាស់មានការចំអកចំអន់លេងថាធ្វើមិនដឹងពីកូនសោឯកជនdអាចត្រូវការកត្តា N ទៅគ្រប់តម្លៃ p និង q ដែលទាំងនេះហួសផុតពីដែនកំណត់នៃការគណនា បច្ចុប្បន្នប្រសិនបើ ចំនួនបឋមទាំងនោះ ធំពេក គឺអាចមានទៅដល់200ខ្ទង់ (របាយការណ៍ពិភពលោកបច្ចុប្បន្នសម្រាប់ផលគុណកត្តាពីរ $N = pq$ ដែល p និង q និមួយៗមានត្រឹម100ខ្ទង់ប៉ុណ្ណោះ)។

ទាំងនេះជាការពិតក្នុងការកំណត់*ថ្នាក់សន្តិសុខនៃគោងការណ៍ (level of security of the scheme)*។ សញ្ញាណសំខាន់អំពីក្រាបសម្ងាត់នៃកូនសោសាធារណៈ*(public-key cryptography)* គឺជាការពិតណាស់ ដែលថាប្រព័ន្ធសម្ងាត់ដែលផុតពីគ្រោះភ័យដ៏រាបណាការគណនាត្រូវការខិតខំប្រឹងប្រែងបំបែកវាឲ្យរួចផុតពីធនធានអ្នកវាយប្រហារ។ ទាំងនេះជាចំនួន *priori* ថេរមួយ (ធម្មតា 2^{128} ឬ2^{256} ជាប្រមាណវិធីចំនួនតូច) និងអាស្រ័យលើបរិបទហើយបំណងនៃការទាក់ទង។ជាការពិតណាស់សារសម្ងាត់ *CIA* និង *email* រវាងអ្នកប្រើប្រាស់ *internet* ពីរនាក់គឺថាសង្ឃឹមនិងខ្នាតគំរូខុសៗគ្នាជាច្រើននៃសន្តិសុខសុវត្ថិភាព!!!

ហត្ថលេខា

យើងអាចត្រឡប់មកបញ្ហាបច្ចុប្បន្នអំពីកម្មវិធីសម័យទំនើប។ នៅក្នុងបញ្ហាពាក្យឯទៀតនៃយថាភាព *(authentication)*។ ហេតុនេះហើយយើងត្រូវមានកូនសោសាធារណៈនិងកូនសៅឯកជន។ ជាក់ស្តែង *Alice* គឺ

គាត់ប្រើកូនសោឯកជនជាការកើតឡើងដោយហត្ថលេខាស្នាមម្រាមដៃ *(digital signature)* ក្នុងការធ្វើ ជាឯកសារ ដើម្បីបង្ហាញយថាភាពរបស់ (ហត្ថលេខាអាស្រ័យលើឯកសារនិងមិនអាច *cut* និង *paste* ទៅឯកសារផ្សេងទេ)។ អ្នកទទួលពេលនោះត្រួតពិនិត្យហត្ថលេខាស្នាមម្រាមដៃដោយការប្រើកូនសៅ សាធារណៈ។

សម្រាប់ឧទាហរណ៍ឧបមាថាកុំព្យូទ័ររបស់អ្នកប្រាប់អ្នកថាវាបង្កើតសម័យទំនើបសម្រាប់ *Adobe* ។ តើការធ្វើកុំព្យូទ័រស្គាល់កម្មវិធីមកពី *Adobe* និងមិនលាក់បាំងមេរោគដូចជាសម័យទំនើបបានដោយរបៀបណា? ដំស្រាយនេះគឺថាសម័យទំនើបមានហត្ថលេខាស្នាមម្រាមដៃ *(digital signature)* ជាមួយការគោរព តាមកូនសៅសាធារណៈ*Adobe*។ កូនសោទាំងនេះពិតណាស់គឺតំឡើងលើកុំព្យូទ័ររបស់អ្នកក្នុងកម្មវិធី *Adobe* ដូច្នោះកុំព្យូទ័រអ្នកអាចត្រួតពិនិត្យហត្ថលេខាមុនការតំឡើងសម័យទំនើបពោលគឺការត្រួតពិនិត្យជោគជ័យបង្ហាញពីសម័យទំនើបជាពិតមកពី *Adobe* និងមិនមួយទៀង។

យើងបង្ហាញបច្ចុប្បន្នពីរបៀបបង្កើតហត្ថលេខាស្នាមម្រាមដៃក្នុងការប្រើ *RSA*។ កូនសៅឯកជន និងកូនសៅសាធារណៈ គឺមានការសរសេរលេខសម្ងាត់ *(encryption)* ដូចគ្នា។ ពេល *Alice* ចង់បាន ឯកសារយថាភាព *(authentic- ate)* (ឧទាហរណ៍៖ កម្មវិធីទំនើបមួយ) គាត់សរសេរអក្សរសម្ងាត់ *(encode)* ជាចំនួនគត់ $0 \prec x \prec N$ ដោយគណនា $\sigma \equiv x^d (\mathrm{mod}\, N)$ និងភ្ជាប់ហត្ថលេខា σ ទៅឯកសារ។ ការត្រួតពិនិត្យ ហត្ថលេខាស្នាមម្រាមដៃសៀវភៅ Bob ដែលបង្កើតដោយកូនសោសាធារណៈរបស់ Alice និងត្រួតពិនិត្យ ស្នាមម្រាមដៃដោយការការគណនា $x' = \sigma^e (\mathrm{mod}\, N)$ និងពិនិត្យថា $x' = x$ ។ គឺវាច្បាស់ណាស់ត្រឹមត្រូវរបស់ប្រព័ន្ធសម្ងាត់ *RSA* ហើយភាពចំអកចំអន់លេងរបស់បំណងអ្នកប្រើប្រាស់ ក្នុងការផលិតហត្ថលេខាដ៏ត្រឹមត្រូវមួយ (សម្រាប់ឧទាហរណ៍អ្នកធ្វើម៉ូតគំនូរមេរោគ) អាចត្រូវការស្វ័យគុណ d ហើយចេញពីនេះត្រូវការកត្តា *N*។

ការស្រាវជ្រាវបច្ចុប្បន្ន

យើងឲ្យព្រាងប្រព័ន្ធសម្ងាត់ *RSA* និងការរៀបចំហត្ថលេខាស្នាមម្រាមដៃ ប៉ុន្តែមានប្រព័ន្ធមូលដ្ឋានគ្រឹះ ជាច្រើនទៀតដែលក្នុងមុខវិជ្ជាគណិតវិទ្យាដូចជា៖កាយរាប់អស់ *(finite fieds)*, ខ្សែកោងអេលីប (elliptic curves), ប្រព័ន្ធនៃសមីការច្រើនអថេរមិនលីអែរ *(systems of non-linear multivariate equations)*, ការពិនិត្យអក្សរសម្ងាត់លំអៀង *(error-correcting codes)* និងច្រើនជាងនេះទៀង។

សាច់រឿងដែលកើតនាពេលអនាគតនៃប្រព័ន្ធសម្ងាត់កូនសោសាធារណៈ***(A post-quantum scenario)*** តើបញ្ហាទាំងអស់របស់សដោះស្រាយដោយប្រព័ន្ធ *RSA* បានឬទេ? ជាអកុសល ចម្លើយនោះ គឺមិនបានទេ។ សន្តិសុខនៃ *RSA* ពោលគឺស្មើគ្នាគោងការណ៍ផ្សេងទៀងជាច្រើន ដែលមានមូលដ្ឋានគ្រឹះក្នុងទ្រឹស្តីចំនួន ពោលគឺគំរាមកំហែងដោយពិតប្រាកដនូវសក្តានុពលអភិវឌ្ឍន៍ពីចំនួន កុំព្យូទ័រប្រព័ន្ធសម្ងាត់។ ម៉្យាងវិញទៀត *algorithm* របស់ *Shor* ត្រូវបានបោះពុម្ពផ្សាយជាសាធារណៈនៅឆ្នាំ 1994 ជាក្រដាសដែលឲ្យប្រផ្នូលអត្ថបទ "ពហុធានៃពេល *algorithm* សម្រាប់លោការីតដាច់ៗពីគ្នា និងកត្តាផលគុណក្នុងកុំព្យូទ័រប្រព័ន្ធសម្ងាត់ "ជាសមត្ថភាពនៃបំបែករបស់ប្រព័ន្ធសម្ងាត់ *RSA* ហើយកុំព្យូទ័រ ប្រព័ន្ធសម្ងាត់អាចបង្កើតចំនួនធំល្មមផងបានដែរ។ កុំព្យូទ័រប្រព័ន្ធសម្ងាត់ *(quantum computers)* សម្រាប់ទំហំ ដែលតូចបំផុតគឺពិតណាស់ហើយនិងវាជាទំនងសង្ឃឹមបានល្អប្រសើរឡើងនាពេលអនាគតឆាប់ៗ។

ដូច្នោះវាជាការសំខាន់ក្នុងផ្តល់ប្រព័ន្ធផ្លាស់គ្នានូវសន្តិសុខសុវត្ថិភាពណាមួយដែលមិនមានឥទ្ធិពលក្នុងករណីសាច់រឿងដែលកើតនាពេលអនាគតក្លាយជាការពិត។សហគមន៍ក្រាបសម្ងាត់ *(The cryptographic community)* គឺមានសកម្មណាស់ក្នុងការចាត់ការនេះនិងការស្រាវជ្រាវបច្ចុប្បន្ន គឺយកចិត្តទុកដាក់ក្នុងការ អភិវឌ្ឍប្រព័ន្ធសម្ងាត់ថ្មីៗមកពីតំបន់ផ្សេងៗនៃគណិតវិទ្យាដែលជាមានការគណនាលេខជាមូលដ្ឋានគ្រឹះនៃបញ្ហាត្រូវមានសង្ឃឹមដោយមិនឈឺចាប់ដូចគ្នាភាពងាយឲ្យឈឺចាប់ទៅនិង *algorithm* ប្រព័ន្ធសម្ងាត់ *(quantum algorithms)* ។