Raport de cercetare pentru UROP

Dragoş Alin Rotaru

Universitatea din Bucuresti, Romania
r.dragos0@gmail.com

Abstract. None

Keywords: securitate, scheme de partajare

1 Introducere

1.1 Istoric

Termenul de criptografie este definit in dictionarul Oxford ca fiind "arta de a scrie si a rezolva coduri". Criptografia moderna s-a desprins de cea clasica in jurul anilor '80, motivand implementarea rigurozitatii matematice pentru definirea constructiilor criptografice. Asta pentru ca in anii anteriori, experienta a dovedit nesiguranta metodelor de criptare, criptanaliza lor fiind uneori triviala (cifrul lui Cezar, Vigenere **??**, **??**) sau uneori atinsa cu ceva mai mult efort precum Enigma si alte metode din cel de-al doilea razboi mondial. **??**

Criptografia moderna se gaseste pretutindeni in viata de zi cu zi de la ATM-uri, cartele telefonice la semnaturi digitale, protocoale de autentificare, licitatii electronice sau bani digitali, luand amploare o data cu aparitia sistemelor cu cheie publica. O definitie potrivita ar fi "studiul stiintific al tehnicililor pentru a securiza informatia digitala, tranzactiile si calculul distribuit.". [1]

1.2 Securitatea Teoretica a Informatiei

2 Scheme de partajare

In cazul unor criptosisteme acestea nu pot fi compromise chiar daca adversarul dispune de o putere computationala nelimitata. Cateva exemple de criptosisteme care garanteaza securitatea teoretica-informationala sunt: schemele de partajare, unele protocoale multi-party computation, preluarea intr-un mod sigur(securizat?) informatii de la baze de date. Securitatea teoretica vine insa cu un cost: efortul computational depus este mult mai mare decat in cazul schemelor care nu garanteaza securitatea teoretica (se bazeaza pe dificultatea computationala unor probleme cunoscute). [2]

O schema de partajare consta in distribuirea unui obiect, o informatie secreta $\mathcal{S}$ la mai multi participanti intr-un mod astfel incat oricare grup predefinit inainte sa poate reconstitui secretul $\mathcal{S}$.

2.1 Constructii existente

Primele scheme de partajare au fost dezvoltate independent de Adi Shamir si George Blakley in 1979. [3, 4] Denumite si scheme de treshold, acestea rezolvau cazul in care oricare grup de participanti cu cardinal ¿= k (dimensiunea thresholdului) puteau reconstitui secretul $\mathcal{S}$ din partile primite de la dealer. Daca schema este perfect sigura atunci oricare grup cu un numar de participanti ¡ k nu obtineau vreo informatie despre $\mathcal{S}$.

Alte scheme de partajare bazandu-se pe grupuri speciale de acces (in cazul schemei lui Shamir, acestea trebuia sa aiba cardinalul ¿= k) au fost dezvoltate de Ito, Saito, si Nishizeki, realizand o generalizare a schemei lui Shamir. [5] Benaloh si Leichter au demonstrat ca schemele de partajare threshold nu pot garanta construirea decat unei fractiuni din multimea functiilor de partajare. Cei doi prezinta un exemplu trivial pentru care schema lui Shamir este insuficienta: consideram cazul in care vrem sa partajam un secret unor 4 participanti: A, B, C, D astfel incat $A + B = \mathcal{S}$ si $C + D = \mathcal{S}$, iar restul de combinatii ale share-urilor sa nu poate reconstitui $\mathcal{S}$ unde cu + notam operatia de reuniune a share-urilor dintre 2 persoane. [6]. Dezavantajul acestor scheme este dimensiunea share-urilor, facand adesea majoritatea constructiilor impracticabile. [7] De asemenea, s-au dezvoltat scheme pentru modele de calcul neconventional, cum ar fi cel cuantic. [8]

Schema lui Shamir

Schema Ito, Saito, si Nishizeki In continuare vom descrie modalitatea de distribuire a share-urilor de la care au pornit Ito, Saito si Nishizeki pentru ca schema sa aiba o structura de acces $\mathcal{A} \subseteq 2^P$ unde P este mulimea de participanti in cadrul procesului. Fie o multime de participanti $P = P_1, P_2, ...P_n$.

- Alegem doua numere intregi k si M, $k \leq M$ si un $q = p^z$ unde p este un numar prim iar z numar intreg pozitiv. Fie $K = GF(q)$
- Alegem $a_1, a_2, ...a_{k-1} \in K - 0$ intr-un mod aleator
- Luam polinomul $f(x) = a_{k-1} * x^{k-1} + a_{k-2} * x^{k-2} + ... + a_1 + \mathcal{S}$.
- Alegem M elemente distincte, $x_1, x_2, \cdots, x_M \in K-0$ si $Q = s_i = f(x_i), 1 \leq m \leq M$
- Alegem $S_i \subseteq Q, 1 \leq i \leq n$ si atribuim fiecarui participant P_i pe S_i. Denumim functia care se ocupa de atribuire $Assign : P \to 2^Q$. [5]

In mod evident, pentru ca modul de atribuire a share-urillor $Assign$ sa respecte structura de acces A atunci $A = \bigcup_i |Assign(i)| \geq k$

3 Sisteme de stocare de lunga durata

In acesta sectiune vom arata cateva intrebuintari ale schemelor de partajare. Consideram cazul in care vrem sa stocham rapoarte medicale, imagini, documente clasificate pe un timp indelungat intr-un mediu electronic. Pe parcursul

timpului, pot apare in schimb, diverse probleme precum dezastre naturale, defectiunea unor componente hardware, eroare umana, etc. [9] Un sistem de stocare necesar nevoilor noastre trebuie sa satisfaca cel putin urmatoarele 3 conditii:

- Disponibilitatea: Informatia trebuie sa ramana accesibila tot timpul, in ciuda erorilor de tip hardware.
- Integritatea: Abilitatea sistemului de a raspunde cererilor intr-un mod care garanteaza corectitudinea lor.
- Confidentialitatea: O persoana care nu face parte din grupul de acces sa nu obtina permisiunea de a afla informatii de orice fel despre datele existente in sistem

Una dintre solutiile existente in a construi acest sistem ar putea fi criptarea datelor insa aceasta nu garanteaza confidentialitatea lor pentru un adversar fara o limita computationala. Majoritatea tehnicilor de criptarea se bazeaza pe dificultatea factorizarii unui numar sau cea a calcularii logaritmului discret insa o data cu dezvoltarea calculatoarelor cuantice aceste probleme nu vor mai fi atat de dificile. [10]

O alternativa la solutia cu criptare care asigura confidentialitatea dar si redundanta necesara este intrebuintarea sistemelor de stocare bazate pe scheme de partajare. [9, 11, 12]

3.1 PASIS

In 2000, PASIS este oferit ca o solutie pentru un sistem descentralizat care ofera beneficii precum securitate, redundanta de date si auto-intretinere. Structurile descentralizate impart informatia la mai multe noduri folosind scheme de redundanta precum RAID pentru a asigura performanta, scalabilitatea sistemului dar si integritatea datelor. [13]

PASIS foloseste schemele de partajare pentru a distribui informatia nodurilor de stocare dintr-o retea. Aceasta presupune folosirea unor agenti pe partea clientului pentru a scrie sau sterge date din noduri. Share-urile obtinute dintr-un fisier sau orice obiecte, sunt puse in retea cu ajutorul agentilor. Pe langa continutul brut al share-urilor se adauga si overhead pentru a retine adresa nodului din retea la care a fost trimisa dar si noul nume cu care este salvata remote.

Considerand o schema de partajare $p - m - n$ unde oricare din cei m participanti pot reconstitui fisierul, dar mai putin de p nu obtin vreo informatie dintr-un total de n participanti. Atunci cand un candidat initiaza o cerere pentru a citi un fisier atunci agentul PASIS aflat local face urmatoarele:

- Cauta numele celor n share-uri care alcatuiesc fisierul intr-un serviciu care listeaza toate datele
- Trimite cereri de a citi fisierul la cel putin m din cele n noduri
- In caz ca acesta nu primeste cel putin m raspunsuri continua pasul anterior incercand query-uri la alte noduri
- Reconstituie fisierul obtinut din cele m share-uri

Operatia de scriere este similara cu cea de citire, aceasta oprindu-se atunci cand pe cel putin $n - m + 1$ noduri s-au scris cu succes share-uri. In articol se mentioneaza si compromisurile de timp/spatiu folosite de PASIS. In schimb, autorii specifica solutii impracticabile pentru auto mentenanta sistemului, considerand ca se poate face prin monitorizare periodica a starii sistemului.

3.2 GridSharing

In 2005 Subbiah si Blough propun o noua abordare pentru a construi un sistem de stocare sigur si tolerant la erori numit GridSharing. Arhitectura este construita pe baza unei scheme de partajare XOR si o multitudine de servere ce folosesc mecanisme de replicare permitandu-i schimbarea parametrilor pentru diferite metrici de evaluare a performantelor. [12]

Articolul identifica 3 tipuri de defectiuni ale serverelor cu ajutorul carora sunt stocate datele:

- Caderi: un server este *cazut* daca nu mai raspunde vreunui mesaj din retea si s-a oprit din a mai efectua vreo operatie
- Bizantine: atunci cand serverul respecta intotdeauna protocoalele initiale. Se considera ca share-urile salvate local au fost compromise
- Leakage-only(scapare?): serverul executa protocoalele corect dar e posibil ca un adversar sa fi obtinut share-urile stocate

Primele 2 modele definite mai sus sunt preluate din calculul cu sisteme distribuite. Cel de-al 3-lea model a fost introdus pentru a defini atacatorul care foloseste vulnerabilitatile cu intentia doar de a invata din date. Arhitectura GridSharing consta in N servere unde cel mult c servere pot cade(? srsly) b server bizantine si l leakage only. Cele N pot fi aranjate intr-un grid cu r linii si N/r coloane. Caracteristicile modelului bizantin si leakage-only permit ca share-urile de pe cel mult $l + b$ linii sa permita dezvaluirea lor unui adeversar.

3.3 POTSHARDS

In 2007 este propus un nou sistem care combina caracteristicile PASIS si GridSharing adaugand posiblitatea de a realiza migrarea datelor la noduri noi. De asemenea este introdusa o tehnica noua de gasire a share-urilor folosind pointeri aproximativi. Pentru a asigura confidentialitatea, autorii adopta o schema de partajare XOR te tipul totul sau nimic, la fel ca in GridSharing. POTSHARDS readreseaza(?) problema in care o persoana neautorizata care incearca sa afle informatii vulnerabile sa nu scape nedetectata. Schemele existente precum PASIS si GridSharing nu indeplineau aceasta cerinta daca un atacator determina locatia share-urilor distribuite. Solutia pe care o ofera PASIS este reconstruirea share-urilor insa aceasta putea sa aiba repercursiuni negative, precum dezvaluirea unor date secrete, POTSHARDS poate fi gandit ca o aplicatie pe partea de client care comunica cu o multime de noduri (arhive) independente . In prima faza, POTSHARDS partajeaza obiectele in fragmente la care adauga meta-date. Autorii le

numesc shard-uri. Shard-urile sunt trimise apoi arhivelor independente, fiecare avand domeniul propriu de securitate. Pentru a reconstitui cu succes obiectele, meta-datele shard-urile contin detalii despre structura pointerilor aproximativi, indicand regiunea in care se afla urmatorul shard. Pentru un atacator, detinerea unui shard nu il ajuta foarte mult, pentru a detecta urmatorul shard, un atac brut force consta in cereri multiple in zona indicata de pointerul aproximativ. Un astfel de atac nu va trece neobservat de POTSHARDS deorece unul dintre scopurile sale este sa stocheze datele intr-un mod cat mai "imprastiat". [9]

4 Criptare VS scheme de partajare

5 Rezultate obtinute

5.1 Arhitectura sistemului

5.2 Erori gasite in articol

5.3 Verificarea rezultatelor

5.4 Publicarea articolului

References

1. Katz, J., Lindell, Y.: Introduction to Modern Cryptography (Chapman & Hall/Crc Cryptography and Network Security Series). Chapman & Hall/CRC (2007)
2. Csirmaz, L.: The size of a share must be large. Journal of cryptology **10.4** (1997) 223–231
3. Blakley, G.: Safeguarding cryptographic keys. Proceedings of the 1979 AFIPS National Computer Conference (1979) 313–317
4. Shamir, A.: How to share a secret. Commun. ACM **22**(11) (1979) 612–613
5. Ito, M., Saito, A., Nishizeki, T.: Secret sharing scheme realizing general access structure. Electronics and Communications in Japan (Part III: Fundamental Electronic Science) **72**(9) (1989) 56–64
6. Benaloh, J., Leichter, J.: Generalized secret sharing and monotone functions. In: Proceedings on Advances in Cryptology. CRYPTO '88, New York, NY, USA, Springer-Verlag New York, Inc. (1990) 27–35
7. Beimel, A.: Secret-sharing schemes: a survey. In: Coding and cryptology. Springer (2011) 11–46
8. Hillery, M., Bužek, V., Berthiaume, A.: Quantum secret sharing. Physical Review A **59**(3) (1999) 1829
9. Storer, M.W., Greenan, K.M., Miller, E.L., Voruganti, K.: Potshards - a secure, recoverable, long-term archival storage system. TOS **5**(2) (2009)
10. Shor, P.W.: Algorithms for quantum computation: discrete logarithms and factoring. In: Foundations of Computer Science, 1994 Proceedings., 35th Annual Symposium on, IEEE (1994) 124–134
11. Wylie, J.J., Bigrigg, M.W., Strunk, J.D., Ganger, G.R., Kiliççöte, H., Khosla, P.K.: Survivable information storage systems. Computer **33**(8) (2000) 61–68
12. Subbiah, A., Blough, D.M.: An approach for fault tolerant and secure data storage in collaborative work environments. In: StorageSS. (2005) 84–93
13. Patterson, D.A., Gibson, G., Katz, R.H.: A case for redundant arrays of inexpensive disks (raid). SIGMOD Rec. **17**(3) (June 1988) 109–116