Mihalea Andrei
341C4

File-injection attacks on searchable encryption

Introducere

Principalul scop al searchable encryption este de a permite unui client sa caute in anumite fisiere criptate de pe un server, pastrand un grad de securitate atat asupra cautarilor clientului cat si asupra fisierelor de pe server. Toate schemele SE au un anumit grad de scurgere a informatiilor, iar cele mai recente cercetari se axeaza pe gasirea unor astfel de scheme care sa fie atat eficiente dar si sa aiba un grad mic de scurgeri de informatii. In principal, cele mai curente scheme SE prezinta scurgeri ale tiparelor de cautare, de exemplu, cand o cautare se repeta si scurgeri cu privire la tiparul accesului la fisiere, adica ce fisiere sunt returnate cand o anumita cautare este facuta. De curand, s-a aratat ca si scurgeri foarte mici de date pot fi folosite pentru a extrage informatii sensibile, mai ales daca un potential atacator are informatii cu privire la anumite cuvinte cheie sau fisiere. In alte lucrari, s-a aratat ca in cazul in care serverul cunoaste continuturile fisierelor clientului, atunci isi poate da seama de cautarile acestuia, prin folosirea celor doua tipuri de scurgeri pomenite mai sus.

Contributia lucrarii

In aceasta lucrare se analizeaza consecintele scurgerilor de date din schemele SE folosind atacuri in care fisiere sunt injectate pe server. In astfel de atacuri, serverul trimite anumite fisiere clientului, care le cripteaza si apoi le incarca, folosind regulile schemei SE. In articol, se arata ca o foarte mare parte din cuvintele cheie introduse de client pot fi aflate folosind un atac cu injectare de fisiere, chiar daca numarul de fisiere injectate nu este unul mare.
Sunt doua tipuri de atacuri: adaptive si neadaptive, un adaptivitatea este data de modul in care se face injectarea. Daca aceasta se face inainte de cautari, atunci atacul este unul neadaptiv, iar daca dupa fiecare cautare se injecteaza fisiere, atunci atacul este adaptiv. Un astfel de atac construieste fisierele injectate dupa observarea scurgerilor de date din cautarile anterioare.

Background

Pe scurt, SE permite clientilor sa incarce versiuni criptate ale unor fisiere pe un server si sa primeasca inapoi acele fisiere care contin anumite cuvinte cheie. Clientul care vrea sa caute un cuvant cheie, mai intai calculeaza determinist un jeton pentru acest cuvant, si apoi il trimite la

server. Serverul afla ce descifreaza jetonul si returneaza fisierele care contin cuvantul cheie respectiv. Scopul unui atac este de a afla cuvantul cheie care corespunde unui jeton.
Pentru atacurile din aceasta lucrare, se considera ca fisierele contin doar cuvinte cheie care se repeta o singura data. Desi in realitate acestea ar putea sa contina si alte cuvinte pe langa cele cheie, iar cele cheie sa se repete de mai multe ori, acest lucru nu este relevant pentru scopurile lucrarii.
Pe parcursul lucrarii, se va presupune K = {k1, k2, …} multimea de cuvinte cheie.
Atacurile adaptive sunt mai eficiente decate cele neadaptive, dar pentru aceasta se presupune ca schemele SE nu satisfac proprietatea de forward privacy, Forward privacy inseamna ca serverul nu poate sa spuna daca un fisier nou inserat corespunde unei cereri anterioare acestui eveniment.

Atac folosind cautare binara

Folosind acest atac, serverul poate afla toate cuvintele cheie pe care un client le cauta, fara sa aiba nicio informatie cu privire la continutul fisierelor. Serverul injecteaza un numar de fisiere egal cu jumatate din numarul cuvintelor cheie.
Vom presupune ca lungimea multimii K este o putere a lui 2, deci vom avea log K fisiere injectate. Al i-lea fisier injectat contine cuvintele cheie care au al i-lea bit 1. Pentru a afla care este cuvantul cheie, parcurgem toate fisierele injectate, pentru cele care sunt in lista fisierelor returnate intersectam continutul, iar pentru cele care nu sunt scadem continutul acestora din rezultatul intersectiilor.

Numar de fisiere injectate: log K
Acuratete: 100%

Acest atac poate fi totusi usor de combatut prin stabilirea unui prag, care va fi denumit de acum incolo T, care sa reprezinte numarul maxim de cuvinte cheie dintr-un fisier. Pentru a combate acest atac, acest prag ar trebui sa fie mai mic decat jumatate din numarul de cuvinte cheie

Atac folosind cautare ierarhica

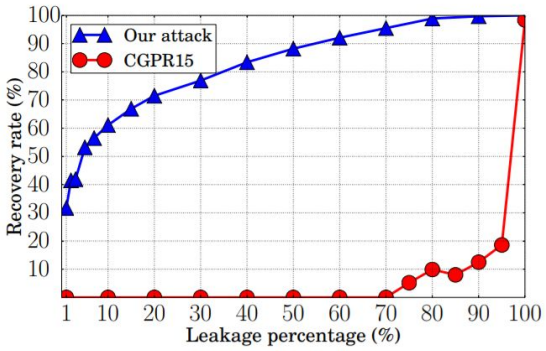
Acest atac are rolul de a combate contramasura folosita impotriva atacului prin cautare binara. Prima oara se partitioneaza universul cuvintelor cheie in subseturi de dimensiune T, apoi pentru fiecare subset injecteaza cate un fisier care contine cuvintele cheie prezente in acesta. Astfel se afla subsetul in care cheia. Serverul mai injecteaza fisiere conform atacului folosind cautare binara, pentru seturi de cuvinte cheie corespunzatoare a cate doua fisiere consecutive. Dupa ce se afla fisierul in care este cheia, se foloseste algoritmul de recuperare a cheii folosit in atacul prin cautare binara pe fisierele injectate formate din cate doua fisiere consecutive.

Numar de fisiere injectate: |K|/2T·(log 2T+2)
Acuratete: 100%

Atac folosind cunostinte partiale

In acest tip de atac se presupune ca serverul cunoaste anumite informatii despre unele fisiere ale clientului. Fisierele despre care serverul are informatii se numesc leaked files. Atacul se bazeaza pe frecventa aparitiilor jetoanelor si cuvintelor cheie in fisierele. Frecventa unui jeton, respectiv cuvant cheie se defineste ca fractiunea din fisierele clientului care contin acel jeton sau cuvant cheie. Serverul poate sa stie frecventa unui jeton, dar nu poate sa stie frecventa exacta a unui cuvant cheie, asa ca se va folosi de frecventa estimata, care se obtine din fisierele scurse (leaked files). Se va presupune K' setul format din 2T cuvinte cheie care au cea mai apropiata frecventa de jetonul cautat. Se va aplica algoritmul de cautare binara folosind ca univers al cheilor K'. Spre deosebire de celelalte metode de pana acum aceasta nu returneaza mereu un rezultat si este unul adaptiv, astfel incat fisierele injectate depind de jetonul cautat.

Numar de fisiere injectate: log 2T
Acuratete: <100%, depinde de leaked files

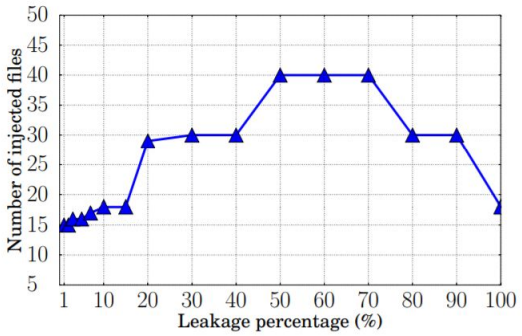


Recuperarea mai multor chei

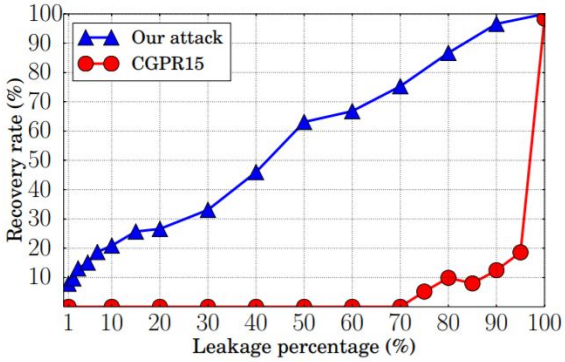
Considerand ca vrem sa recuperam cheile pentru m jetoane, am putea sa aplicam de m ori metoda descrisa mai sus, gasind pentru fiecare jeton 2T cele mai apropiate chei ca frecventa estimata si apoi sa se reuneasca aceste subesturi. Problema este ca este aproape sigur ca vor rezulta mai mult de 2T cuvinte cheie. Astfel, serverul poate creea un univers candidat de lungime 2T/m pentru fiecare jeton si reuninuea acestora va avea lungime 2T. Totusi, daca m este mare, toate subseturile vor fi mici si rata cu care cheile vor fi recuperate va fi foarte mica.

Se propune un atac format din doi pasi principali. In prima faza, se vor recupera cheile pentru un subset de lungime n cu n << m. Acest set de jetoane cu cuvintele cheie corespunzatoare se numeste ground truth. Al doilea pas realizeaza recuperarea restului cheilor. Acest pas se bazeaza pe fatpul ca daca stim jetonul t, atunci pentru un alt jeton t', pe care vrem sa-l aflam frecventa perechii (t, t') ar trebui sa fie apropiata de frecventa estimata (k, k'), unde k' este presupusul cuvant cheie corespunzator lui t'. Parametrul δ determina cat de apropiate sunt aceste frecvente si acesta se determina statistic. Frecventa estimata pentru (k, k') trebuie sa fie intre +- δ inmultit cu aceasta frecventa din valoarea frecventei perechii (t, t') pentru cel putin 99% din cazuri. Daca subsetul rezultat in acest pas este suficient de mic, se va aplica atacul prin cautare binara, iar daca nu, cel ierarhic.

Numar fisiere injectate: depinde



Acuratete: < 100%



Cereri cu doua cuvinte cheie

Presupunem ca avem cheile cautate k1 si k2 deci vom partitiona universul K in doua subseturi K1 si K2 astfel incat k1 se va afla in K1 si k2 se va afla in K2. Pentru a face asta, generam o secventa de lungime log K formata din perechi de subseturi ($K1^i$, $K2^i$). $K1^i$ va fi subsetul care contine toate cuvintele care au pe bitul i valoarea 0, iar $K2^i$ va fi subsetul complementar. Prima oara se vor genera fisiere care contin cuvintele cheie din $K1^i$ si respectiv din $K2^i$. Se vor genera apoi fisiere folosind atacul prin cautare binara cu subsetul K1, iar apoi in aceste fisiere se vor introduce toate cuvintele cheie din K2 si invers.

Pentru a recupera cheile, se va parcurge primul set de fisiere si se va vedea daca niciuna dintre chei nu se gaseste in fisiere. Vom gasi aceste fisiere $F1^i$ si $F2^i$, si vom aplica algoritmul de recuperare al atacului prin cautare binara pe fisierele obtinute conform acestui atac corespunzatoare lui $F1^i$ si $F2^i$.

Numar de fisiere injectate: $\log^2 |K|+\log|K|$
Acuratete: <100%

Recuperarea cheilor din cereri cu mai multe chei

Atacul din sectiunea anterioara functioneaza pentru o cerere conjunctiva formata din doua chei, utilizand $\log^2 |K|+\log|K|$ fisiere injectate. Se propune o metoda care foloseste $O(\log |K|)$ si recupereaza un numar oricat de mare de chei.
Ideea atacului este de a recupera cheile una cate una, pornind de la cea mai mare in ordine lexicografica. Prima oara, serverul injecteaza primele $|K|/2$ chei intr-un singur fisier, apoi exista doua posibilitati: daca toate cuvintele cheie se afla in acest fisier, atunci se vor injecta primele $|K|/4$ cuvinte cheie. Daca fisierul nu este returnat, atunci inseamna ca cel putin un cuvant cheie nu se afla in acesta, deci se injecteaza un fisier care contine primele $3|K|/4$ cuvinte cheie. Astfel, serverul afla cuvantul cel mai mare din punct de vedere al ordinii lexicografice. Se aplica acelasi algoritm pentru a afla restul cuvintelor cheie, doar ca cele deja gasite vor fi incluse in fisierul injectat.

Numar fisiere injectate: $d \log|K|$, unde d este numarul de jetoane din cerere
Acuratete: 100%

Implementare

Am considerat n o putere a lui doi chei, respectiv jetoane. Cheile reprezinta forma binara a numerelor de la 0 la n - 1, iar pentru a nu mai calcula jetoanele, am considerat ca acestea sunt exact numerele de la 0 la n - 1, deoarece scopul lucrarii nu este de a implementa o schema SE. Cand un client cere un jeton, serverul ia respectivul cuvant cheie corespunzator jetonlui si returneaza fisierele care contin acel cuvant cheie. In realitate, o schema SE nu cunoaste cuvintele cheie corespunzatoare unui jeton. Aceasta primeste jetonul si calculeaza care este cheia. Totusi, pentru simplitate am considerat ca atunci cand trimit un jeton, serverul stie ce cuvant cheie sa caute in fisiere. Totusi acest lucru nu influenteaza modul in care un atac este efectuat.
Am implementat toate cele 6 atacuri si sunt functionale.

Concluzie

Atacurile prin injectare de fisiere pot fi devastatoare impotriva schemelor SE. In lucrare se demonstreaza ca scurgerile de date sunt periculoase pentru confidentialitatea datelor clientului si chiar niste scurgeri relativ mici de informatii pot sa dezvaluie informatii sensibile despre acestea.