# **Count Attacks against Searchable Symmetric Encryption**

## Case study I

d1 = {w1, w2, w3, w4, w6}

d2 = {w1, w2, w3, w6},

d3 = {w1, w2, w4, w6},

d4 = {w1, w4, w6},

d5 = {w4, w5, w6}, d6 = {w4, w6},

**Step 1:** create inverted index of the breached data and create co-occurrence matrix for the breached dataset.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| keyword | Document id | | | | | | frequency |
| w1 | d1 | d2 | d3 | d4 |  |  | 4 |
| w2 | d1 | d2 | d3 |  |  |  | 3 |
| w3 | d1 | d2 |  |  |  |  | 2 |
| w4 | d1 |  | d3 | d4 | d5 | d6 | 5 |
| w5 |  |  |  |  | d5 |  | 1 |
| w6 | d1 | d2 | d3 | d4 | d5 | d6 | 6 |

Breached data:

Keyword Co-occurrence matrix:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | w1 | w2 | w3 | w4 | w5 | w6 |
| w1 | 4 | 3 | 2 | 3 | 0 | 4 |
| w2 | 3 | 3 | 2 | 2 | 0 | 3 |
| w3 | 2 | 2 | 2 | 1 | 0 | 2 |
| w4 | 3 | 2 | 1 | 5 | 1 | 5 |
| w5 | 0 | 0 | 0 | 1 | 1 | 1 |
| w6 | 4 | 3 | 2 | 5 | 1 | 6 |

**Step 2:** Construct the co-occurrence matric for the query response from the server.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Query | eid | | | | | | frequency |
| q1 | e1 | e2 | e3 |  | e5 | e6 | 5 |
| q2 |  |  | e3 | e4 |  |  | 2 |
| q3 | e1 | e2 | e3 | e4 |  |  | 4 |
| q4 |  | e2 | e3 | e4 |  |  | 3 |
| q5 | e1 | e2 | e3 | e4 | e5 | e6 | 6 |
| q6 |  |  |  |  |  | e6 | 1 |

Server response:

Query Co-occurrence matrix:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | q1 | q2 | q3 | q4 | q5 | q6 |
| q1 | 5 | 1 | 3 | 2 | 5 | 1 |
| q2 | 1 | 2 | 2 | 2 | 2 | 0 |
| q3 | 3 | 2 | 4 | 3 | 4 | 0 |
| q4 | 2 | 2 | 3 | 3 | 3 | 0 |
| q5 | 5 | 2 | 4 | 3 | 6 | 1 |
| q6 | 1 | 0 | 0 | 0 | 1 | 1 |

**Step3:** Create the mapping knowledge K, which includes the mapping between query tokens and keywords that share a unique frequency of occurrences.

Unique length: {q3: w1}, {q4: w2}, {q2: w3}, {q1: w4}, {q6: w5}, {q5: w6}

Since, all the unique keyword mapping established for query and keyword is recovered. The count attack is possible.

## 

## Case study II

d1 = {w1, w2, w3, w4, w6},

d2 = {w2, w3, w6},

d3 = {w1, w3, w4, w6},

d4 = {w1, w4, w6},

d5 = {w4, w5, w6},

d6 = {w5},

**Step 1:** create inverted index of the breached data and create co-occurrence matrix for the breached dataset.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| keyword | Document id | | | | | | frequency |
| w1 | d1 |  | d3 | d4 |  |  | 3 |
| w2 | d1 | d2 |  |  |  |  | 2 |
| w3 | d1 | d2 | d3 |  |  |  | 3 |
| w4 | d1 |  | d3 | d4 | d5 |  | 4 |
| w5 |  |  |  |  | d5 | d6 | 2 |
| w6 | d1 | d2 | d3 | d4 | d5 |  | 5 |

Breached data:

## Keyword Co-occurrence matrix:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | w1 | w2 | w3 | w4 | w5 | w6 |
| w1 | 3 | 1 | 2 | 3 | 0 | 3 |
| w2 | 1 | 2 | 2 | 1 | 0 | 2 |
| w3 | 2 | 2 | 3 | 2 | 0 | 3 |
| w4 | 3 | 1 | 2 | 4 | 1 | 4 |
| w5 | 0 | 0 | 0 | 1 | 2 | 1 |
| w6 | 3 | 2 | 3 | 4 | 1 | 5 |

## Step 2: Construct the co-occurrence matric for the query response from the server.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Query | eid | | | | | | frequency |
| q1 |  | e2 | e3 | e4 | e5 | e6 | 5 |
| q2 |  | e2 | e3 |  | e5 | e6 | 4 |
| q3 | e1 |  | e3 |  |  |  | 2 |
| q4 |  |  |  | e4 | e5 |  | 2 |
| q5 |  | e2 |  |  | e5 | e6 | 3 |
| q6 |  |  |  | e4 | e5 | e6 | 3 |

## Server response:

## 

## Query Co-occurrence matrix:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | q1 | q2 | q3 | q4 | q5 | q6 |
| q1 | 5 | 4 | 1 | 2 | 3 | 3 |
| q2 | 4 | 4 | 1 | 1 | 3 | 2 |
| q3 | 1 | 1 | 2 | 0 | 0 | 0 |
| q4 | 2 | 1 | 0 | 2 | 1 | 2 |
| q5 | 3 | 3 | 0 | 1 | 3 | 2 |
| q6 | 3 | 2 | 0 | 2 | 2 | 3 |

**Step3:** Create the mapping knowledge K, which includes the mapping between query tokens and keywords that share a unique frequency of occurrences.

## Unique length: {q1: w6}, {q2: w4}

## No unique length: {q3, q4: w2, w5}, {q5, q6: w1, w3}

## Step4: Examine the co-occurrence matrix to identify potential candidate keywords for the remaining query tokens and their occurrences.

## For {q3, q4: w2, w5} and using unique length: {q1: w6},

## (q3, q1) = 1 (w2, w6) = 2 -> not possible

## (q3, q1) = 1 (w5, w6) = 1 -> possible

## (q4, q1) = 2 (w2, w6) = 2 -> possible

## (q4, q1) = 2 (w5, w6) = 1 -> not possible

## Extracted data: {q3: w5}, {q4: w2}

## For {q3, q4: w2, w5} and using unique length: {q2: w4}

## (q3, q2) = 1 (w2, w4) = 1 -> possible

## (q3,q2) = 1 (w5,w4) = 1 -> possible

## (q4, q2) = 1 (w2, w4) = 1 -> possible

## (q4, q2) = 1 (w5, w4) = 1 -> possible

## Extracted data: {q3: w2,w5}, {q4:w2,w5} -> still not possible

## For {q5, q6: w1, w3} and using unique length: {q2: w4}

## (q5, q2) = 3 (w1, w4) = 3 -> possible

## (q5, q2) = 3 (w3, w4) = 2 -> not possible

## (q6, q2) = 2 (w1, w4) = 3 -> not possible

## (q6, q2) = 2 (w3, w4) = 2 -> possible

## Extracted data: {q5: w1}, {q6: w3}

Since, all the unique keyword mapping established for query and keyword is recovered. The count attack is possible.

Keyword mapping:

Unique length: {q1: w6}, {q2: w4}, {q3: w5}, {q4: w2}, {q5: w1}, {q6: w3}

## 

## Case study III

d1 = {w1, w2, w3, w6},

d2 = {w2, w3, w4, w6},

d3 = {w1, w2, w3, w4, w6},

d4 = {w1, w3, w4, w6},

d5 = {w5, w6},

d6 = {w5, w6}

**Step 1:** create inverted index of the breached data and create co-occurrence matrix for the breached dataset.

Breached data:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| keyword | Document id | | | | | | frequency |
| w1 | d1 |  | d3 | d4 |  |  | 3 |
| w2 | d1 | d2 | d3 |  |  |  | 3 |
| w3 | d1 | d2 | d3 | d4 |  |  | 4 |
| w4 |  | d2 | d3 | d4 |  |  | 3 |
| w5 |  |  |  |  | d5 | d6 | 2 |
| w6 | d1 | d2 | d3 | d4 | d5 | d6 | 6 |

Keyword Co-occurrence matrix:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | w1 | w2 | w3 | w4 | w5 | w6 |
| w1 | 3 | 2 | 3 | 2 | 0 | 3 |
| w2 | 2 | 3 | 3 | 2 | 0 | 3 |
| w3 | 3 | 3 | 4 | 3 | 0 | 4 |
| w4 | 2 | 2 | 3 | 3 | 0 | 3 |
| w5 | 0 | 0 | 0 | 0 | 2 | 2 |
| w6 | 3 | 3 | 4 | 3 | 2 | 6 |

## Step2: Construct the co-occurrence matric for the query response from the server.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Query | eid | | | | | | frequency |
| q1 | e1 |  |  |  | e5 | e6 | 3 |
| q2 | e1 | e2 |  |  |  | e6 | 3 |
| q3 |  |  | e3 | e4 |  |  | 2 |
| q4 | e1 | e2 | e3 | e4 | e5 | e6 | 6 |
| q5 |  | e2 |  |  | e5 | e6 | 3 |
| q6 | e1 | e2 |  |  | e5 | e6 | 4 |

## Server response:

## 

## Query Co-occurrence matrix:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | q1 | q2 | q3 | q4 | q5 | q6 |
| q1 | 3 | 2 | 0 | 3 | 2 | 3 |
| q2 | 2 | 3 | 0 | 3 | 2 | 3 |
| q3 | 0 | 0 | 2 | 2 | 0 | 0 |
| q4 | 3 | 3 | 2 | 6 | 3 | 4 |
| q5 | 2 | 2 | 0 | 3 | 3 | 3 |
| q6 | 3 | 3 | 0 | 4 | 3 | 4 |

## 

**Step3:** Create the mapping knowledge K, which includes the mapping between query tokens and keywords that share a unique frequency of occurrences.

## Unique length: {q4: w6}, {q6: w3}, {q3: w5}

## No unique length: {q1, q2, q5: w1, w2, w4}

## Step4: Examine the co-occurrence matrix to identify potential candidate keywords for the remaining query tokens and their occurrences.

## For {q1, q2, q5: w1, w2, w4} and using unique length: {q4: w6}

## (q1, q4) = 3 (w1, w6) = 3 -> possible

## (q2, q4) = 3 (w2, w6) = 3 -> possible

## (q5, q4) = 3 (w4, w6) = 3 -> possible

## For {q1, q2, q5: w2, w1, w4} and using unique length: {q4: w6}

## (q1, q4) = 3 (w2, w6) = 3 -> possible

## (q2, q4) = 3 (w1, w6) = 3 -> possible

## (q5, q4) = 3 (w4, w6) = 3 -> possible

## For {q1, q2, q5: w4, w2, w4} and using unique length: {q4: w6}

## (q1, q4) = 3 (w4, w6) = 3 -> possible

## (q2, q4) = 3 (w2, w6) = 3 -> possible

## (q5, q4) = 3 (w1, w6) = 3 -> possible

## Extracted data: {q1, q2, q5: w1, w2, w4} -> still not possible to extract.

## For {q1, q2, q5: w1, w2, w4} and using unique length: {q6: w3}

## (q1, q6) = 3 (w1, w3) = 3 -> possible

## (q1, q6) = 3 (w2, w3) = 3 -> possible

## (q1, q6) = 3 (w4, w3) = 3 -> possible

## (q2, q6) = 3 (w1, w3) = 3 -> possible

## (q2, q6) = 3 (w2, w3) = 3 -> possible

## (q2, q6) = 3 (w4, w3) = 3 -> possible

## (q5, q6) = 3 (w1, w3) = 3 -> possible

## (q5, q6) = 3 (w2, w3) = 3 -> possible

## (q5, q6) = 3 (w4, w3) = 3 -> possible

## Extracted data: {q1, q2, q5: w1, w2, w4} -> still not possible to extract.

## For {q1, q2, q5: w1, w2, w4} and using unique length: {q3: w5}

## (q1, q3) = 0 (w1, w5) = 0 -> possible

## (q1, q3) = 0 (w2, w5) = 0 -> possible

## (q1, q3) = 0 (w4, w5) = 0 -> possible

## (q2, q3) = 0 (w1, w5) = 0 -> possible

## (q2, q3) = 0 (w2, w5) = 0 -> possible

## (q2, q3) = 0 (w4, w5) = 0 -> possible

## (q5, q3) = 0 (w1, w5) = 0 -> possible

## (q5, q3) = 0 (w2, w5) = 0 -> possible

## (q5, q3) = 0 (w4, w5) = 0 -> possible

## Extracted data: {q1, q2, q5: w1, w2, w4} -> still not possible to extract.

## The following can’t be mapped {q1, q2, q5: w1, w2, w4} since, the q1, q2, q5 and w1, w2, w4 share equal probability of combination.

## The count attack is not possible for the case study 3, only part of the keyword can be extracted which is as follows:

## Unique Length: {q4: w6}, {q6: w3}, {q3: w5}