Conjunctive Keyword Search

Lorenz Minder

lorenz.minder@epfl.ch

LMA, EPFL

The Problem

- $(V_{i,j})_{1 \le i \le n, 1 \le j \le m}$: database containing n records. A record has m keyword fields.
- Stored on an *untrusted* server.
- Need to be able to *search*: Find the is such that

$$V_{i,j_1} = k_{j_1} \wedge \cdots \wedge V_{i,j_\ell} = k_{j_\ell}.$$

The UNIX-password approach

Idea: Use a distinct hashing function for each keyword field separately.

m	number of keyword fields
$h_1, \ldots, h_m$	hashing functions

Encryption: $(k_1, \ldots, k_m) \mapsto (h_1(k_1), \ldots, h_m(k_m))$.

Search:

- Query $k_{j_1} = w_1 \wedge \cdots \wedge k_{j_\ell} = w_\ell$ transmitted as $(\ell, (j_1, \ldots, j_\ell), (h_{j_1}(w_1), \ldots, h_{j_\ell}(w_\ell)))$.
- Server checks for each file if $h_{j_i}(k_i) = h_{j_i}(w_i)$ for $i = 1, \ldots, \ell$.

The drawback

The server can construct queries itself.

Condition: A search is *secure*, if a server can only deduce logic combinations of requested queries.

The (simpler) GSW scheme

Basis: Decisional Diffie Hellman problem.

- $G = \langle \alpha \rangle$, group generated by α.
- $V_{i,j} \in \mathbb{Z}_q$, where $q = |G|$.
- $a_i \in \mathbb{Z}_q$, chosen randomly ($i = 1, \ldots, n$)
- Encrypt the ith message as

$$(\alpha^{a_i}, \alpha^{a_i V_{i,1}}, \ldots, \alpha^{a_i V_{i,m}}).$$

Queries in the GSW scheme

Query: $V_{i,j_1} = k_1 \wedge \cdots \wedge V_{i,j_\ell} = k_\ell$,

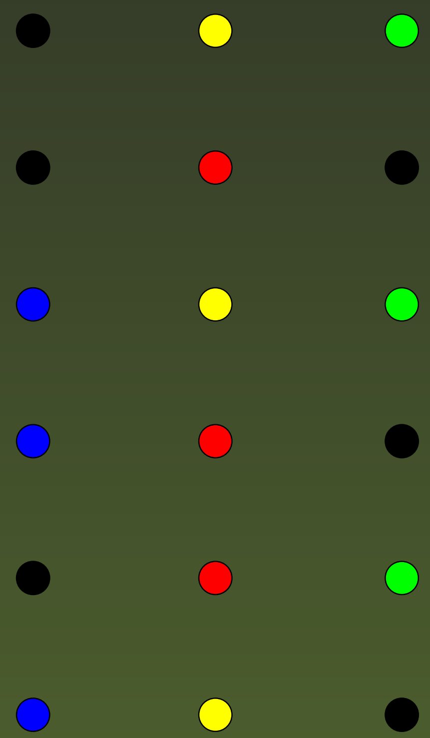
- *Proto*-part of size $O(n)$: $s \in_R \mathbb{Z}_q$; transmit
$$Q := \left(\alpha^{a_1 s}, \alpha^{a_2 s}, \ldots, \alpha^{a_n s}\right).$$
- *Request*-part: Let $C := s + \sum_{w=1}^{\ell} k_w$ Transmit
$$\left(C, \{j_1, \ldots, j_\ell\}\right).$$
- *Verify* by checking if $\alpha^{a_i C} \cdot (\alpha^{\sum_{w=1}^{\ell} V_{i,j_w}})^{-1} = \alpha^{a_i s}$.

Security of GSW

- Proofs based on *hardness of DH* for G.
- *Leak*: Server knows $\{j_1, \ldots, j_\ell\}$ for every query.

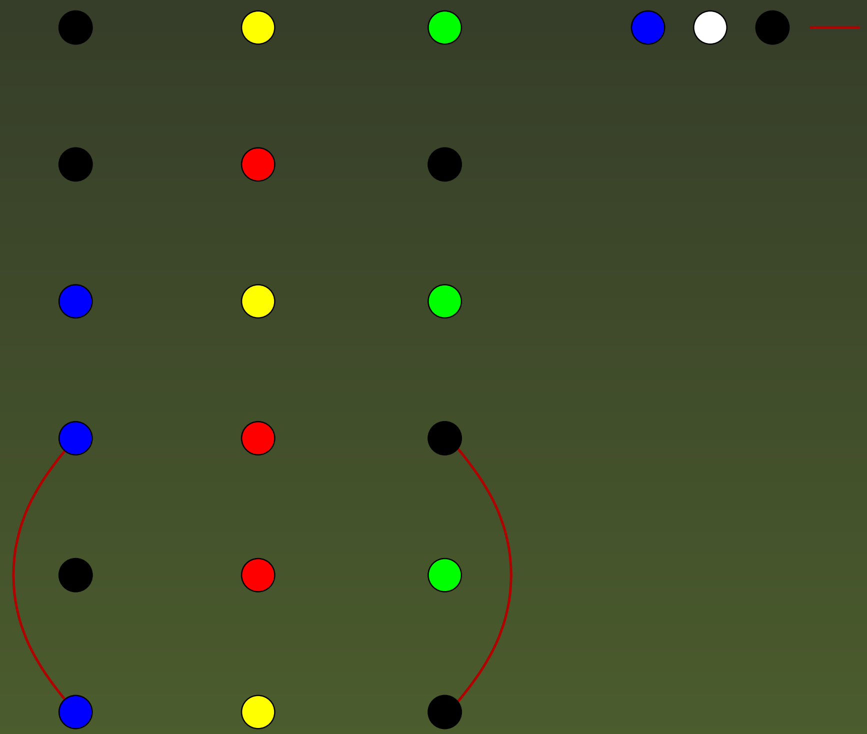
Exploiting the leak

Idea: Build a graph collecting knowledge.



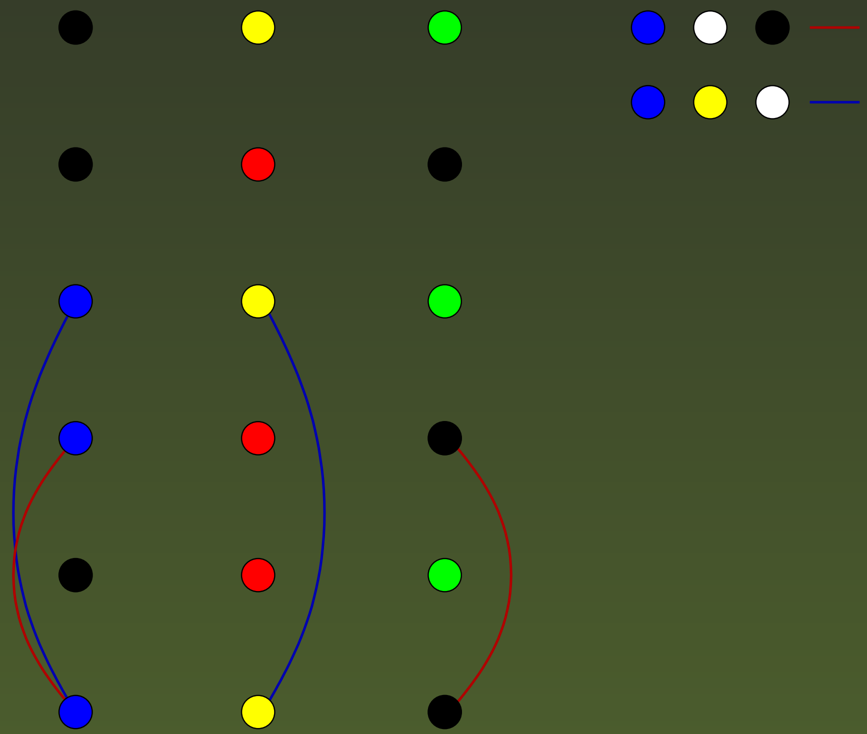
Exploiting the leak

Idea: Build a graph collecting knowledge.



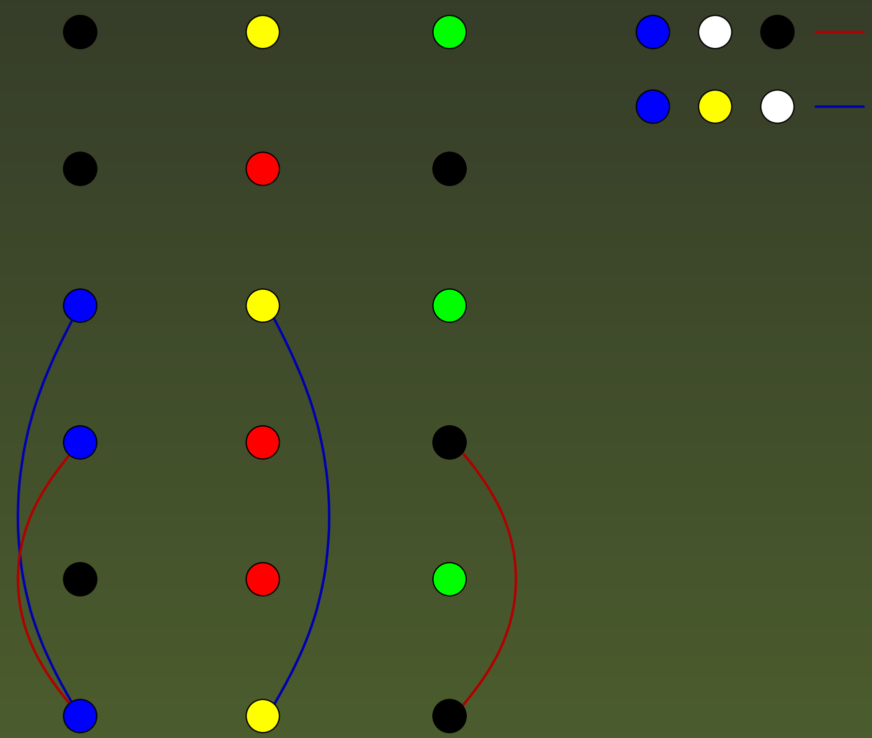
Exploiting the leak

Idea: Build a graph collecting knowledge.



Exploiting the leak

Idea: Build a graph collecting knowledge.



Look at connected components!

How well does it work?

- Effect: Reduction to UNIX-password scheme.

How well does it work?

- Effect: Reduction to UNIX-password scheme.
- Single keyword queries must be avoided.

How well does it work?

- Effect: Reduction to UNIX-password scheme.
- Single keyword queries must be avoided.
- The success depends on the distribution of keywords and queries.

How well does it work?

- Effect: Reduction to UNIX-password scheme.
- Single keyword queries must be avoided.
- The success depends on the distribution of keywords and queries.
- Simple model: Need $O(n \log n)$ matches.

Larger Example (Simulation)

Database with 100000 entries, search for 2 keywords.

```
total # of queries 210260
empty returns = 173677 (82.6011%)
useless returns (0 or 1 match)= 192468 (91.5381%)
cumulative # query results was 218801
adjusted cumulative # query results was 200010
field 0:  largest is 15000, largest contained comp is 9592
field 1:  largest is 20000, largest contained comp is 14573
field 2:  largest is 10000, largest contained comp is 175
field 3:  largest is 16600, largest contained comp is 11668
field 4:  largest is 12500, largest contained comp is 1887
```

Possible remedies

- Disallow single keyword queries.

Possible remedies

- Disallow single keyword queries.
- Obfuscation with artificial keyword fields.

Possible remedies

- Disallow single keyword queries.
- Obfuscation with artificial keyword fields.
- Periodically reencrypt, permuting the entries.

Part II

Different approach:

Coding based schemes

RS codes

$[n,k]_q$ RS code: $\alpha_1,\ldots,\alpha_n \in \mathbb{F}_q$, encode:

$$\mathbb{F}_q[X]_{<k} \longrightarrow \mathbb{F}_q{}^n$$
$$(f_0,\ldots,f_{k-1}) \mapsto (f(\alpha_1),\ldots,f(\alpha_n)).$$

We write d the minimum distance, and e the correction bound.

The BW-decoder for RS codes

$(y_1, \ldots, y_n)$: received erroneous codeword.

- Find $g \in \mathbb{F}_q[X]_{<k+e}$ and $h \in \mathbb{F}_q[X]_{<e+1}$ such that

$$g(\alpha_i) = y_i \cdot h(\alpha_i), \qquad i = 1, \ldots, n.$$

The BW-decoder for RS codes

$(y_1, \ldots, y_n)$: received erroneous codeword.

- Find $g \in \mathbb{F}_q[X]_{<k+e}$ and $h \in \mathbb{F}_q[X]_{<e+1}$ such that

$$g(\alpha_i) = y_i \cdot h(\alpha_i), \qquad i = 1, \ldots, n.$$

- If the number of errors is small enough, then

The BW-decoder for RS codes

$(y_1, \ldots, y_n)$: received erroneous codeword.

- Find $g \in \mathbb{F}_q[X]_{<k+e}$ and $h \in \mathbb{F}_q[X]_{<e+1}$ such that

$$g(\alpha_i) = y_i \cdot h(\alpha_i), \qquad i = 1, \ldots, n.$$

- If the number of errors is small enough, then
  - ith position in error $\implies h(\alpha_i) = 0.$

The BW-decoder for RS codes

$(y_1, \ldots, y_n)$: received erroneous codeword.

- Find $g \in \mathbb{F}_q[X]_{<k+e}$ and $h \in \mathbb{F}_q[X]_{<e+1}$ such that

$$g(\alpha_i) = y_i \cdot h(\alpha_i), \qquad i = 1, \ldots, n.$$

- If the number of errors is small enough, then
  - ith position in error $\implies h(\alpha_i) = 0$.
  - $f = g/h$.

The BW-decoder for RS codes

To find g and h, solve the linear system

$$V_{k+e}g = DV_{e+1}h,$$

where

$$V_\ell := \begin{pmatrix} 1 & \alpha_1 & \cdots & \alpha_1^{\ell-1} \\ 1 & \alpha_2 & \cdots & \alpha_2^{\ell-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha_n & \cdots & \alpha_n^{\ell-1} \end{pmatrix} \text{ and } D := \begin{pmatrix} y_1 & & & \\ & y_2 & & \\ & & \ddots & \\ & & & y_n \end{pmatrix}.$$

The kernel of the BW-matrix

So, need to determine (an element in) the kernel of

$$A := \left[V_{k+e} \middle| -DV_{e+1} \right].$$

- With the number of errors increasing, the size of the kernel goes down.

The kernel of the BW-matrix

So, need to determine (an element in) the kernel of

$$A := \left[V_{k+e} \middle| -DV_{e+1}\right].$$

- With the number of errors increasing, the size of the kernel goes down.
- If the number of errors goes above the decoding bound, usually the kernel is trivial. Stray solutions are controllable via the choice of e.

Measuring set resemblance using d_H

$S \subset \mathbb{F}_q^n$, random set of size m, $S = \{s_0, \ldots, s_{m-1}\}$.
R. v. $W(S)$: Pick $I_i \in_R \{0, \ldots, m-1\}$, iid. Set

$$W(S) := (s_{I_1,1}, s_{I_2,2}, \ldots, s_{I_n,n}).$$

Measuring set resemblance using d_H

$S \subset \mathbb{F}_q^n$, random set of size m, $S = \{s_0, \ldots, s_{m-1}\}$.
R. v. $W(S)$: Pick $I_i \in_R \{0, \ldots, m-1\}$, iid. Set

$$W(S) := (s_{I_1,1}, s_{I_2,2}, \ldots, s_{I_n,n}).$$

2	2	10	11	9	4	8	12	2	5	7	9	1	6	3	s_0
11	10	11	1	2	9	5	5	1	8	10	2	3	12	5	s_1
3	2	13	9	6	2	7	1	8	5	2	6	9	10	1	s_2
11															W(S)

Measuring set resemblance using d_H

$S \subset \mathbb{F}_q^n$, random set of size m, $S = \{s_0, \dots, s_{m-1}\}$.
R. v. $W(S)$: Pick $I_i \in_R \{0, \dots, m-1\}$, iid. Set

$$W(S) := (s_{I_1,1}, s_{I_2,2}, \dots, s_{I_n,n}).$$

Measuring set resemblance using d_H

$S \subset \mathbb{F}_q^n$, random set of size m, $S = \{s_0, \ldots, s_{m-1}\}$.
R. v. $W(S)$: Pick $I_i \in_R \{0, \ldots, m-1\}$, iid. Set

$$W(S) := (s_{I_1,1}, s_{I_2,2}, \ldots, s_{I_n,n}).$$

2	2	10	11	9	4	8	12	2	5	7	9	1	6	3	s_0
11	10	11	1	2	9	5	5	1	8	10	2	3	12	5	s_1
3	2	13	9	6	2	7	1	8	5	2	6	9	10	1	s_2
11	2	13	11	2	4	5	5	2	8	2	6	9	6	3	W(S)

Set resemblance with d_H (cont'd)

S as before, T another such set, $|T| = \ell$. If $|T \cap S| = t$,

$$\mathrm{E}[\#\text{ zeros in } W(S) - W(T)] = n\left(\frac{t}{m\ell}(1 - q^{-1}) + q^{-1}\right).$$

So,

- Distance of $W(S) - W(T)$ to the zero codeword estimates of the number of matches of S and T.

Set resemblance with d_H (cont'd)

S as before, T another such set, $|T| = \ell$. If $|T \cap S| = t$,

$$\mathrm{E}[\#\text{ zeros in } W(S) - W(T)] = n\left(\frac{t}{m\ell}(1 - q^{-1}) + q^{-1}\right).$$

So,

- Distance of $W(S) - W(T)$ to the zero codeword estimates of the number of matches of S and T.
- Problem: Cannot get close enough to zero!

Dedicated correct positions

One possibility: Fix a few *dedicated correct positions* (dcp).

- A dcp is not subject to the random process, where instead the difference will always be zero.

Dedicated correct positions

One possibility: Fix a few *dedicated correct positions* (dcp).

- A dcp is not subject to the random process, where instead the difference will always be zero.
- If we want t matches to be good enough, need c dcp, where

$$c + (n - c)\frac{t}{m\ell} > \frac{n + k}{2}.$$

Dedicated correct positions

One possibility: Fix a few *dedicated correct positions* (dcp).

- A dcp is not subject to the random process, where instead the difference will always be zero.
- If we want t matches to be good enough, need c dcp, where
$$c + (n-c)\frac{t}{m\ell} > \frac{n+k}{2}.$$
- dcp are an artifact: For some codes, they are not needed.

Modified setting for CKW

Generalization of the conjunctive keyword search problem.

- No keyword fields.

Modified setting for CKW

Generalization of the conjunctive keyword search problem.

- No keyword fields.
- Document = Encrypted set of keywords D.

Modified setting for CKW

Generalization of the conjunctive keyword search problem.

- No keyword fields.
- Document = Encrypted set of keywords D.
- Query = Encrypted set of keywords Q.

Modified setting for CKW

Generalization of the conjunctive keyword search problem.

- No keyword fields.
- Document = Encrypted set of keywords D.
- Query = Encrypted set of keywords Q.
- The server can $\mathrm{checkinter}(D, Q)$, to see if the intersection of D and Q is significant.

Message encryption

A is $n \times n$ random invertible (secret). The jth record has the cryptogram

$$AD_jV_{e+1}S_j,$$

where S_j is $(e+1) \times (e+1)$ random invertible, D_j is diagonal, containing the information and dcp, all entries nonzero. (At the ith dcp, store 1.)

Query encryption

Encryption of a query:

$$A\tilde{D}V_{k+e}T,$$

T is $k+e \times k+e$ random invertible (one-time).

Construction of $\tilde{D}$:

- Pick a random codeword $(c_1, \ldots, c_n)$.

$$\tilde{D}_{ii} = \begin{cases} c_i^{-1} & \text{for a dcp,} \\ (c_i y_i)^{-1} & \text{otherwise.} \end{cases}$$

Verification

The server counts the number of solutions to

$$\begin{aligned} & A\tilde{D}V_{k+e}T\tilde{g} = AD_jV_{e+1}S_j\tilde{h} \\ \iff \quad & V_{k+e}T\tilde{g} = \tilde{D}^{-1}D_jV_{e+1}S_j\tilde{h} \end{aligned}$$

to see if the jth document matches.

Note that $\tilde{D}^{-1}D_j$ is diagonal, with entries equal to c_i on dcp and on matching positions.

$\implies$ the number of solutions gives an indication on the quality of the match.

Security considerations

- Right scramblers haven't been broken in > 20 years.

Security considerations

- Right scramblers haven't been broken in > 20 years.
- Use AG codes, not RS codes.

Security considerations

- Right scramblers haven't been broken in > 20 years.
- Use AG codes, not RS codes.
- What about left scramblers?

Left scramblers

It's a question of ongoing work.

- Do not know reduction of hard problems from this.

Left scramblers

It's a question of ongoing work.

- Do not know reduction of hard problems from this.
- Unaware of a good algorithm breaking it.

Left scramblers

It’s a question of ongoing work.

- Do not know reduction of hard problems from this.
- Unaware of a good algorithm breaking it.

If the left scrambler is *defeated*:

- Might possibily result in similar attacks as the one presented against the GSW-scheme.

Main points

- A query should not reveal *which* keyword is queried.

Main points

- A query should not reveal *which* keyword is queried.
- Codes are promising candidates for the CKW problem.

Main points

- A query should not reveal *which* keyword is queried.
- Codes are promising candidates for the CKW problem.
  - The Hamming metric fits well in this context.

Main points

- A query should not reveal *which* keyword is queried.
- Codes are promising candidates for the CKW problem.
  - The Hamming metric fits well in this context.
  - Possible extensions: Keyword weighting, fuzzy search.

Main points

- A query should not reveal *which* keyword is queried.
- Codes are promising candidates for the CKW problem.
  - The Hamming metric fits well in this context.
  - Possible extensions: Keyword weighting, fuzzy search.
- More theoretical insight on the left scramblers is needed.

Main points

- A query should not reveal *which* keyword is queried.
- Codes are promising candidates for the CKW problem.
  - The Hamming metric fits well in this context.
  - Possible extensions: Keyword weighting, fuzzy search.
- More theoretical insight on the left scramblers is needed.
- Complexities are polynomial in the parameters, but currently too large for practical uses.

Main points

- A query should not reveal *which* keyword is queried.
- Codes are promising candidates for the CKW problem.
  - The Hamming metric fits well in this context.
  - Possible extensions: Keyword weighting, fuzzy search.
- More theoretical insight on the left scramblers is needed.
- Complexities are polynomial in the parameters, but currently too large for practical uses.
- Search time is $O(N)$ instead of $O(\log(N))$.

References

- Golle, P.; Waters, B.; Staddon, J. *Secure conjunctive keyword search over encrypted data*. Proceedings of the Second International Conference on Applied Cryptography and Network Security (ACNS-2004); 2004 June 8-11; Yellow Mountain, China. Heidelberg: Springer-Verlag; 2004; Lecture Notes in Computer Science 3089: 31-45.