Near-collisions and their Impact on Biometric Security

Axel DURBET[1], Paul-Marie GROLLEMUND[2], Pascal LAFOURCADE[1] and Kevin THIRY-ATIGHEHCHI [1]

[1]Université Clermont-Auvergne, CNRS, Mines de Saint-Étienne, LIMOS, France
[2]Université Clermont-Auvergne, CNRS, LMBP, France

SECRYPT
July 12, 2022

Table of Contents

Biometric Terminology

Biometric System

Biometric data:

- Biological or physical characteristic: fingerprint, DNA, iris, . . .
- The collected data are in a metric space.

Enrollment:

- Provide a biometric data, which will be altered and used as a reference.
- Potentially provide a second factor (*e.g.* password, token, . . .).

Authentication:

- Provide a fresh biometric data and an optional a second factor.
- Comparison with the reference data.
- If the difference is smaller than a threshold ϵ, the authentication is a success.

Biometric System

Biometric data:

- Biological or physical characteristic: fingerprint, DNA, iris, ...
- The collected data are in a metric space.

Enrollment:

- Provide a biometric data, which will be altered and used as a reference.
- Potentially provide a second factor (*e.g.* password, token, ...).

Authentication:

- Provide a fresh biometric data and an optional a second factor.
- Comparison with the reference data.
- If the difference is smaller than a threshold ϵ, the authentication is a success.

Biometric System

Biometric data:

- Biological or physical characteristic: fingerprint, DNA, iris, ...
- The collected data are in a metric space.

Enrollment:

- Provide a biometric data, which will be altered and used as a reference.
- Potentially provide a second factor (*e.g.* password, token, ...).

Authentication:

- Provide a fresh biometric data and an optional a second factor.
- Comparison with the reference data.
- If the difference is smaller than a threshold ϵ, the authentication is a success.

Feature and Template

Feature:

- A feature is a characteristic information of the biometric data.
- Denoted by $F = E(I)$, where E corresponds to the extraction.

Example: fingerprint minutiae, ...

Template:

- Altered (protected) version of the feature.
- Denoted by $T = \mathcal{T}(P, F) \in \mathbb{F}_2^n$, where P is a token and F a feature.

Feature and Template

Feature:

- A feature is a characteristic information of the biometric data.
- Denoted by $F = E(I)$, where E corresponds to the extraction.

Example: fingerprint minutiae, ...

Template:

- Altered (protected) version of the feature.
- Denoted by $T = \mathcal{T}(P, F) \in \mathbb{F}_2^n$, where P is a token and F a feature.

Hypothesis

In this framework, we suppose that:

- Templates are uniformly distributed in $\mathbb{F}_2^n$.
- There exists a reasonable attack for impersonate one user but unreasonable on a whole database.

Problematic

Notations

D_1 : Leaked database.

D_2 : Another database.

Goal:

Find D_2 such that an attacker can impersonate users of D_1.

If the following inequality is fulfilled:

$$|D_2| \leq |D_1|$$

Database Partitionning in Theory

Hierarchical Agglomerative Clustering (HAC)

Definition (Hierarchical Agglomerative Clustering (HAC))

This algorithm takes as input D a template database and s an integer and returns Cls a partition of D such that $\forall\ a, b \in C_i, max(d_H(a, b)) \leq s$.

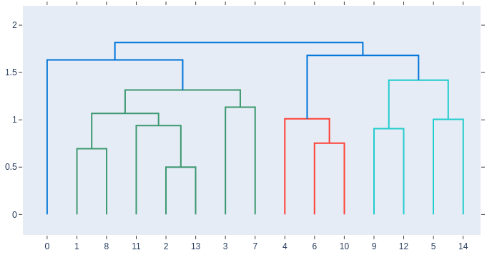


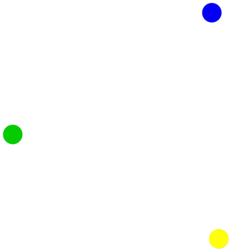
Figure: HAC example.

Master Template

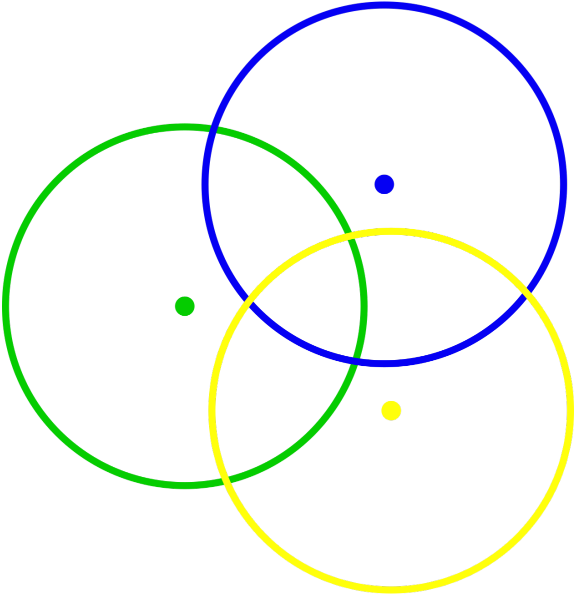
Definition (ϵ-master-template or ϵ-MT)

Let (Ω, d) be the template space and D a template database. A template $t \in \Omega$ is an ϵ-master-template if $\forall\ t' \in D, d(t, t') \leq \epsilon$.

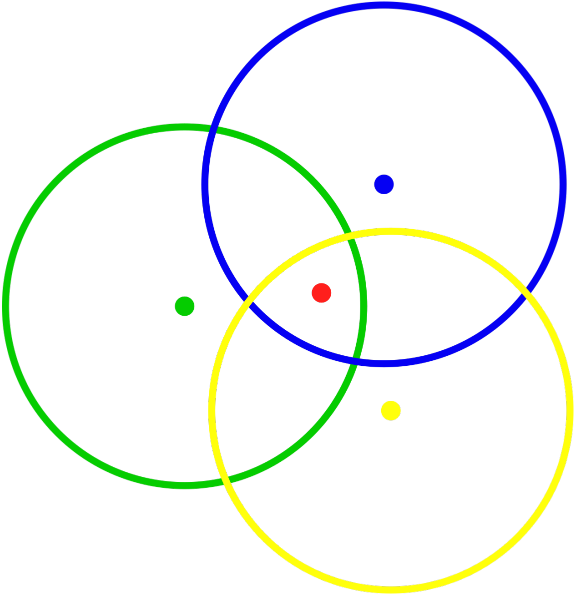
Master Template



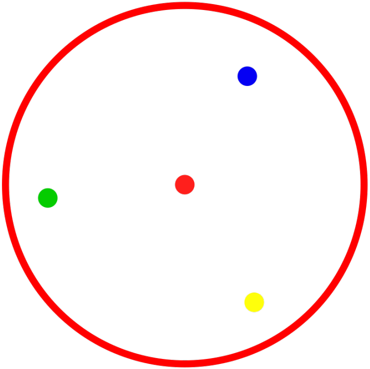
Master Template



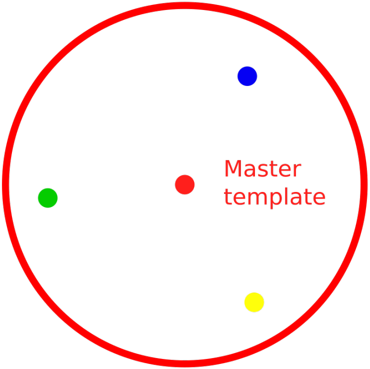
Master Template



Master Template



Master Template

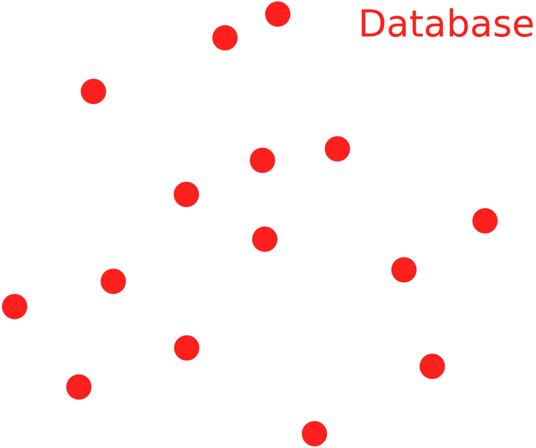


Database Partitionning Algorithm

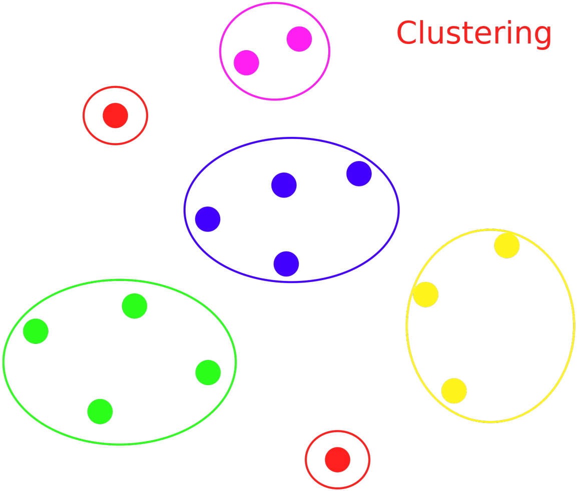
Algorithm 1: Database partitioning algorithm

```
Data: D, ϵ
Result: MTS
Set s to 2ϵ.
Set MTS to [ ].
while D ≠ ∅ do
    Compute cluster Cls using D and s.
    foreach cluster c in Cls do
        Search the cover template t for c.
        if a cover template t is found for c ∈ C then
            Set D to D\c and add t to MTS.
        end
        Set s to s − 1.
    end
end
return MTS.
```

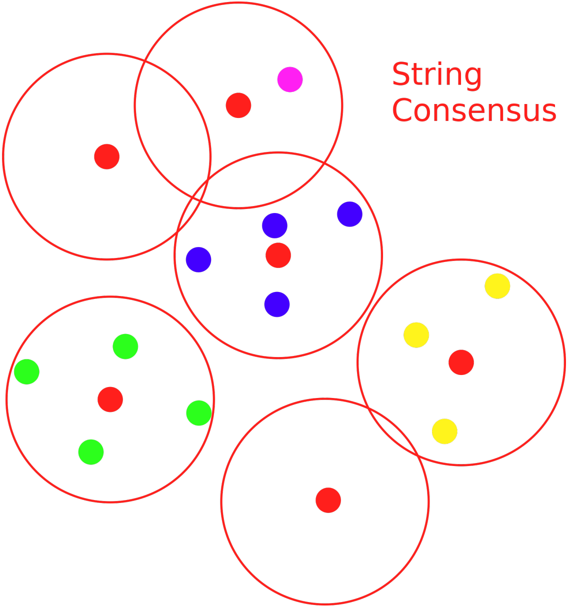
Procedure Illustration



Procedure Illustration



Procedure Illustration



Procedure Illustration

Database Partitionning in Practice

Closest String Problem

Definition (Closest-String Problem)

Given $S = \{s_1, s_2, \ldots, s_m\}$ *a set of strings with length* n*, find a center string* t *of length* m *minimizing* d *such that for every string* s *in* S, $d_H(s, t) \leq d$.

Definition (Modified Closest-String Problem)

Given $S = \{s_1, s_2, \ldots, s_m\}$ *a set of strings with length* n *and* d *a distance, find a center string* t *of length* m *such that for every string* s *in* S, $d_H(s, t) \leq d$.

Theorem (MCSP is NP-hard)

The modified closest-string problem is *NP*-hard.

How to solve MCSP problem ?

Formulating an IP

Solve the following IP (Integer Program) with k the number of targeted clients and v_i their templates:

$$\begin{cases} d_H(p, v_1) \leq \epsilon \\ \vdots \\ d_H(p, v_k) \leq \epsilon \end{cases}$$

System Reduction Theorem

Theorem (System Reduction)

For a given template database D and for a given $v \in D$, consider $L = \{p \in \mathbb{F}_2^n \,|\, AN \leq \epsilon - d(v)\}$ with $N = n_v^I$, $\epsilon = (\epsilon, \dots, \epsilon)^T$,$n_{v,i}$ denotes $d_{K_i}(p, v)$, n_v^I denotes the parameters vector $(n_{v,1}, \dots, n_{v,|I|})$ and $A = (a_{i,j})$ a matrix of size $|I| \times |D|$ whose the $(i,j)^{\text{th}}$ element is

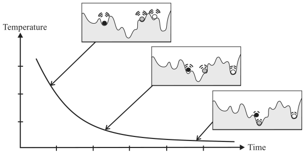
$$a_{i,j} = \begin{cases} 1 & \text{if } d_{K_j}(v_1, v_i) = 0 \\ -1 & \text{if } d_{K_j}(v_1, v_i) = |K_j| \end{cases}$$

Then, $L = \mathcal{C}$ the ϵ-cover-template-set for D.

SANN

The Simulated ANNealing (SANN) is an optimization algorithm which relies on the following parameters:

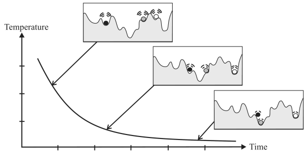
- *Space:*
  $\mathcal{N} = \prod_{k=1}^{|I|} \{0, \ldots, \min(\epsilon, |K_k|)\}$
- *Energy:*
  $E(N) = \sum_{i=1}^{|I|} f((\epsilon - d(v) - AN)_i)$
  with $f(x) = \min(0, x)$.
- *Cooling Schedule:* Linear decreasing temperature.
- *Proposal distribution:* The neighbors set.
- *Termination:* Reaches the maximum iteration number, or if a solution is found.



SANN

The Simulated ANNealing (SANN) is an optimization algorithm which relies on the following parameters:

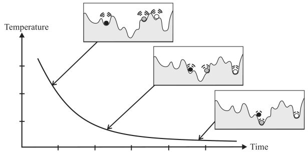
- *Space:* $\mathcal{N} = \prod_{k=1}^{|I|}\{0, \ldots, \min(\epsilon, |K_k|)\}$
- *Energy:* $E(N) = \sum_{i=1}^{|I|} f((\epsilon - d(v) - AN)_i)$ with $f(x) = \min(0, x)$.
- *Cooling Schedule:* Linear decreasing temperature.
- *Proposal distribution:* The neighbors set.
- *Termination:* Reaches the maximum iteration number, or if a solution is found.



SANN

The Simulated ANNealing (SANN) is an optimization algorithm which relies on the following parameters:

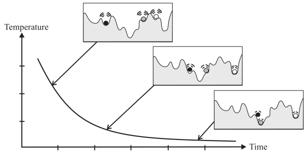
- *Space:* $\mathcal{N} = \prod_{k=1}^{|I|}\{0, \ldots, \min(\epsilon, |K_k|)\}$
- *Energy:* $E(N) = \sum_{i=1}^{|I|} f((\epsilon - d(v) - AN)_i)$ with $f(x) = \min(0, x)$.
- *Cooling Schedule:* Linear decreasing temperature.
- *Proposal distribution:* The neighbors set.
- *Termination:* Reaches the maximum iteration number, or if a solution is found.



SANN

The Simulated ANNealing (SANN) is an optimization algorithm which relies on the following parameters:

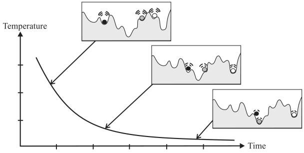
- *Space:* $\mathcal{N} = \prod_{k=1}^{|I|}\{0, \dots, \min(\epsilon, |K_k|)\}$
- *Energy:* $E(N) = \sum_{i=1}^{|I|} f((\epsilon - d(v) - AN)_i)$ with $f(x) = \min(0, x)$.
- *Cooling Schedule:* Linear decreasing temperature.
- *Proposal distribution:* The neighbors set.
- *Termination:* Reaches the maximum iteration number, or if a solution is found.



SANN

The Simulated ANNealing (SANN) is an optimization algorithm which relies on the following parameters:

- *Space:* $\mathcal{N} = \prod_{k=1}^{|I|} \{0, \ldots, \min(\epsilon, |K_k|)\}$
- *Energy:* $E(N) = \sum_{i=1}^{|I|} f((\epsilon - d(v) - AN)_i)$ with $f(x) = \min(0, x)$.
- *Cooling Schedule:* Linear decreasing temperature.
- *Proposal distribution:* The neighbors set.
- *Termination:* Reaches the maximum iteration number, or if a solution is found.



Performance

n	ϵ	#clients	Time (ms)
20			1592
30	10	50	2428
40			3887

n	ϵ	#clients	Time (ms)
	5		24949
70	15	200	20978
	25		29089

n	ϵ	#clients	Time (ms)
		90	11087
70	10	130	18330
		170	20887

Figure: IP approach performance.

n	ϵ	#clients	Error in %	Time (ms)
20			0.64	17
30	10	50	0.00	1
40			0.05	1

n	ϵ	#clients	Error in %	Time (ms)
	5		0.00	36
70	15	200	0.00	36
	25		0.00	40

n	ϵ	#clients	Error in %	Time (ms)
		90	0.14	12
70	10	130	0.00	22
		170	0.00	31

Figure: Stochastic approach performance.

Approach Comparisons

IP approach

Strengths:

- No error possible.
- Easy to set up.

Weaknesses:

- Slow.

Stochastic approach

Strengths:

- Fast.

Weaknesses:

- Could miss a master template.
- Hard to set up.

Global Performance

n	ϵ	#clients	#clust	#clust(G)	Efficiency	Time (ms)	Time G (ms)
20	10	50	2.700	35.433	×13.12	8 415.270	10.714
30			8.709	48.977	× 5.70	8 775.802	18.940
40			18.087	49.986	× 2.77	6 417.596	23.762
70	5	200	200.000	200.000	× 1.00	43.969	449.166
	15		90.000	200.000	× 2.22	47 016.050	337.082
	25		22.109	198.982	× 9.00	222 386.614	346.420
50	10	90	89.67	90	×1.00	136.572	137.186
		130	129.30	130		428.885	251.221
		170	168.79	170		531.363	434.727

$$\text{Efficiency} = \frac{\#\text{clust(G)}}{\#\text{clust}}$$

Global Performance

n	ϵ	#clients	#clust	#clust(G)	Efficiency	Time (ms)	Time G (ms)
20	10	50	2.700	35.433	×13.12	8 415.270	10.714
30			8.709	48.977	× 5.70	8 775.802	18.940
40			18.087	49.986	× 2.77	6 417.596	23.762
70	5	200	200.000	200.000	× 1.00	43.969	449.166
	15		90.000	200.000	× 2.22	47 016.050	337.082
	25		22.109	198.982	× 9.00	222 386.614	346.420
50	10	90	89.67	90	×1.00	136.572	137.186
		130	129.30	130		428.885	251.221
		170	168.79	170		531.363	434.727

$$\text{Efficiency} = \frac{\#\text{clust(G)}}{\#\text{clust}}$$

Security Bound

Near Collision

Definition (Near collision)

Let (Ω, d) be the template space and a threshold ϵ. There exists a near-collision if $\exists a, b \in \Omega \mid d(a, b) \leq \epsilon$.

You said gain?

Definition (Gain)

The gain of the attacker is $G = |D_1| - |D_2|$ *with* D_1 *the leaked database and* D_2 *the construct database.*

You said gain?

Definition (Gain)

The gain of the attacker is $G = |D_1| - |D_2|$ with D_1 the leaked database and D_2 the construct database.

How can we maximise the gain?

- Templates should be as close as possible to each other.

How can we minimise the gain?

- Templates should be as far apart as possible.

The number of near collisions is a good indicator of the expected gain.

0 Gain

How to ensure that the attacker gain is 0?

Birthday Problem

To prevent near collisions, with n the size of a template, the number k of templates which give a collision with a probability of 50% is

$$\approx 2^{n/2} \left(\sum_{i=0}^{\epsilon} \binom{n}{i} \right)^{-1/2}$$

Security Threshold

n	ϵ	$log_2 k$ with 50% near collision
128	12	38
	25	20
256	25	72
	51	38
512	51	139
	102	74
1024	102	276
	204	146

Conclusion

Conclusion

Work done:

- Two solutions for the Near String Problem.
- Method to find a second database (D_2) that the attacker could attack to impersonate all users of a leaked database (D_1) with the constraint that $|D_2| \leq |D_1|$.
- Security bound over the size of a biometric database.

Future work:

- Improving the SANN based method.
- Improving the IP based method.
- Exploring other approaches.

Conclusion

Work done:

- Two solutions for the Near String Problem.
- Method to find a second database (D_2) that the attacker could attack to impersonate all users of a leaked database (D_1) with the constraint that $|D_2| \leq |D_1|$.
- Security bound over the size of a biometric database.

Future work:

- Improving the SANN based method.
- Improving the IP based method.
- Exploring other approaches.

Question time

$$E = m \times C^2$$

$$\text{Energy} = \text{milk} \times \text{Coffee}^2$$

Any Questions ?